

CAPTURING PRODUCT COMPLEMENTARITY IN ASSORTMENT OPTIMIZATION

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Venus Hiu Ling Lo

May 2019

© 2019 Venus Hiu Ling Lo
ALL RIGHTS RESERVED

CAPTURING PRODUCT COMPLEMENTARITY IN ASSORTMENT OPTIMIZATION

Venus Hiu Ling Lo, Ph.D.

Cornell University 2019

We study three assortment optimization problems that reflect the current retail environment. The goal of these problems is to compute an assortment which maximizes the retailer's expected revenue or profit, depending on the scenario. For each problem, we present efficient algorithms to compute the optimal assortment, or prove that the problem is NP-hard and provide an approximation algorithm. First, we study assortment optimization for an omnichannel retailer, who operates both an online store and a physical store offline. The retailer selects a subset of products from the online store to offer in the physical store, knowing that customers can visit the physical store to observe products and update preferences for similar products offered online. We introduce a features tree to organize products by their shared features, and describe how customers update their preferences for online products. Second, we study assortment optimization when certain products look more attractive if they are offered alongside in the assortment. For example, by observing a lower quality product, customers may perceive an increase in value for a target product. We call this effect synergy and introduce a synergy graph to describe the synergy structure between products. We classify the difficulty of the synergistic assortment optimization problem based on the synergy graph. Finally, we study assortment optimization with dynamic substitution and inventory stocking costs. This problem considers customers who arrive over a finite selling horizon to

make a purchasing decision amongst the products that have not stocked-out. The retailer does not have the ability to modify the assortment seen by each arriving customer. Hence, the assortment that each customer purchases from depends on the initial assortment and stocking decisions of the retailer, as well as the purchasing decisions of the preceding customers.

BIOGRAPHICAL SKETCH

Venus was born in Hong Kong and grew up in Toronto. She received her Bachelor of Mathematics in chartered accountancy at the University of Waterloo, followed by masters degrees in accounting and optimization. She decided that optimization is more fun and came to ORIE.

Dedicated to my family.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Professor Huseyin Topaloglu. Huseyin has been very good at pushing me to aim higher when I was succeeding and being patient with me when I struggled with my research. His encouragements gave me the confidence to seek my own projects and take pride in my work. I would also like to thank the ORIE faculty. As part of my minor committee, Professors Kris Iyer and Sid Banerjee have given me great research ideas to pursue. I would also like to thank Sid, along with Professors David Williamson and David Goldberg, for being generous with career advice.

The students of ORIE is a highlight of being at Cornell. I would like to thank Emily Fischer and Matthew Zalesak for listening when I rambled about ideas. Matthew, Ken Chong, James Dong, and David Eckman taught me to bake and cook, and provided edible examples of what I should aim for. Woo-Hyung Cho is probably the nicest student in ORIE, apart from David. Cory Girard, Weici Hu, and Andrew Chee provided me with energy in the form of exotic snacks from Asia. Andrew Daw and Sam Gutekunst taught me how to be a president/dictator of ORGA. XY Han and Chamsi Hssaine suffered through apartment issues with me. Ben Grimmer, Pamela Badian-Pessot, Alyf Janmohamed, and Sean Sinclair made life a blast.

The front office staff is always extremely helpful. In particular, a big thank you to Tara Woodard, who dealt with my numerous last-minute room bookings when I needed to call Huseyin. Katrina Overton has been awesome with helping me navigate through graduation paperwork.

I would also like to acknowledge the Natural Sciences and Engineering Research Council of Canada for their fellowship support.

Thank you to all my friends in the Cornell International Christian Fellowship

and First Ithaca Chinese Christian Church for their prayers and support, and for making five years in Ithaca pass quite pleasantly.

Most importantly, I thank my parents for being supportive throughout all these years. My parents waited (somewhat) patiently for me to finish university, and there were times when there seemed to be no end in sight. And I would not dare forget to include my very annoying brother.

I thank God for giving me this opportunity at Cornell. I never expected to do a PhD, but “ ‘My thoughts are not your thoughts, neither are your ways My ways,’ declares the Lord” (Isaiah 55:8). Life is full of surprises.

TABLE OF CONTENTS

Biographical Sketch	iii
Dedication	iv
Acknowledgements	v
Table of Contents	vii
List of Tables	x
List of Figures	xi
1 Introduction	1
1.1 Assortment Optimization	1
1.2 Overview of Choice Models	3
1.2.1 Multinomial Logit Model	3
1.2.2 Extensions to MNL	7
1.2.3 Non-Parametric Choice Models	9
1.3 Future of Assortment Optimization	11
1.3.1 Omnichannel Assortment Optimization	12
1.3.2 Assortment Optimization in the Presence of Synergy	14
1.3.3 Assortment Optimization with Dynamic Substitution and Stocking Costs	16
1.4 Organization and notation	19
2 Omnichannel Assortment Optimization under the Multinomial Logit Model with a Features Tree	21
2.1 Introduction	21
2.2 The Model	29
2.3 Showroom Setting	34
2.4 General Setting	38
2.4.1 FPTAS for General Setting	39
2.4.2 Solving the Parametrized Problem at a Grid-point	45
2.5 Upper Bound on Optimal Expected Revenue	49
2.6 Extensions	53
2.6.1 Related Online and In-store Assortments	53
2.6.2 Independent Online and In-store Assortments	55
2.7 Numerical Study: Modeling Power of Features Tree Model	56
2.7.1 Ground-Truth and Benchmark Models	57
2.7.2 Test of Predictive Ability	58
2.7.3 Test of Assortments Selected by Features Tree	64
2.8 Numerical Study: Performance of FPTAS	68
2.8.1 Setup	68
2.8.2 Results	69
2.9 Conclusion	70

3	Assortment Optimization under the Multinomial Logit Model with Product Synergies	74
3.1	Introduction	74
3.2	The Model	79
3.2.1	Model and Notations	79
3.2.2	The Parametrized Problem	81
3.3	Optimal Assortments on Synergy Paths and Trees	82
3.3.1	A Synergy Path	83
3.3.2	A Synergy Tree	85
3.4	Sales-Based Linear Program for Synergy Path	88
3.5	Conclusion	91
4	Assortment Optimization with Dynamic Substitution and Inventory Stocking Costs	93
4.1	Introduction	93
4.2	The Model	104
4.3	Dynamic Substitution in the Deterministic Setting	106
4.3.1	NP-hardness of the Setting	107
4.3.2	Product Discontinuation Times vs Stocking Levels	108
4.3.3	K -Choosy Customers	110
4.3.4	Customers with Interval Preference Lists	114
4.4	Dynamic Substitution in the Stochastic Setting	117
4.4.1	Relationship Between the Deterministic and the Stochastic Settings	117
4.4.2	Heuristic: Best-of-the-Best	120
4.4.3	Upper-bound on Expected Profit over Sample Space	121
4.5	Numerical Study	122
4.5.1	Generating Products	123
4.5.2	Generating Customer Arrivals Rates for Customers	124
4.5.3	Structure of the Numerical Experiments	127
4.5.4	Performance of the 2-Approximation Algorithm for Choosy Customers	127
4.5.5	Performance of the Heuristic for Stochastic Setting	128
4.6	Conclusion	138
5	Concluding Remarks	140
A	Appendix: Omnichannel Assortment Optimization under the Multinomial Logit Model with a Features Tree	141
A.1	Proofs Omitted from Chapter 2	141
A.2	Faster Algorithm via Longest Path in a Directed Acyclic Graph	144
A.2.1	Showroom Setting	145
A.2.2	General Setting	148
A.3	Ground-Truth and Benchmark Models	152

A.3.1	Ground-Truth Model	152
A.3.2	Benchmark Model	153
B	Appendix: Assortment Optimization under the Multinomial Logit Model with Product Synergies	155
B.1	Graphs with Low Treewidth	155
B.2	Constructing the Sales-based Linear Program	163
B.3	Proofs Omitted from Chapter 3	170
C	Appendix: Assortment Optimization with Dynamic Substitution and Inventory Stocking Costs	174
C.1	Proofs Omitted from Chapter 4	174

LIST OF TABLES

2.1	Mean absolute error in purchase probability - Features tree and benchmark model	62
2.2	Percentage of optimal expected revenue - Features tree and benchmark model	66
2.3	Performance of FPTAS - General setting	71
4.1	Choosy Customer - Heuristic vs LP upper-bound	131
4.2	Choosy Customer - Heuristic vs naïve rounding	132
4.3	Customers with interval preference lists - Heuristic vs LP upper-bound	136
4.4	Customers with interval preference lists - Heuristic vs naïve rounding	137

LIST OF FIGURES

2.1	Features tree of five phones	23
2.2	Comparison of error - Approximating with features tree	63
2.3	Comparison of revenue - Approximating with features tree	67
3.1	Synergy graph on five products	82
3.2	Example of tree decomposition	87
A.1	Building DAG recursively - Leaf vertices	146
A.2	Building DAG recursively - Non-leaf vertices	146
A.3	Building DAG recursively - Source and sink vertices	147

CHAPTER 1

INTRODUCTION

1.1 Assortment Optimization

One of the most important problems faced by a retailer is deciding on the right set of products to offer to customers. Regardless of whether we are considering physical and digital products, or services like hotel bookings, retailers want to satisfy customers with their products while earning as much revenue as possible. If the retailer offers too few products, then a customer may not find a product that she likes, and can choose to leave without a purchase. As a result, the retailer would lose market share. This effect is also known as spoilage. If the retailer offers too many products, then a customer who would be willing to purchase a higher-margin product can now substitute to a lower-margin product. This effect is also known as product cannibalization or spill-over. There is a clear trade-off between gaining market share and reducing cannibalization effects on products which earn high revenues. The set of products offered by the retailer is called an assortment in the revenue management literature. The problem of finding the assortment that maximizes the retailer's expected revenue is called assortment optimization.

In its most basic form, assortment optimization has two ingredients. The first is the universe of products, $N = \{1, \dots, n\}$. Each product $i \in N$ earns the retailer a revenue of π_i when it is sold. The retailer chooses an assortment, $S \subseteq N$, to offer in his store. The second ingredient of assortment optimization is the customer choice model, which describes how a customer makes purchasing decisions. A customer arrives to the store and either chooses to purchase product $i \in S$, or to

leave the store empty-handed. For each product $i \in S$, the choice model tells us the probability that a customer purchases product i , denoted by $P_i(S)$. Thus, the expected revenue from offering assortment S is $\sum_{i \in S} \pi_i P_i(S)$. The assortment optimization problem is to select an assortment to maximize the retailer's expected revenue:

$$\max_{S \subseteq N} \sum_{i \in S} \pi_i P_i(S). \quad (1.1)$$

In some settings, we may use π_i to denote the profit of product i instead. This distinction is useful in two scenarios. First, we can use this model to consider products with negative profit. It may be beneficial to offer certain products with negative profit if the net effect on profit is positive. For example, a retailer may offer free samples to increase the probability that customers purchase a product rather than walk out empty-handed (Lammers (1991)). Second, the retailer may incur stocking costs if we consider an assortment optimization problem with inventory considerations. He would incur stocking costs without necessarily earning the related revenue.

The model in Problem (1.1) is not very useful from an optimization perspective, because we need to state purchase probabilities under an exponential number of assortments. One solution is to parametrize the purchase probabilities, so that all values of $P_i(S)$ can be computed based on a polynomial number of parameters. The parametrization technique and its interpretation depends on the choice model being studied. The well-known multinomial logit model is an example of a parametrized model. Alternatively, we can use a non-parametrized, ranking-based choice model, such that we represent different customer types by preference rankings over the products in N . To make the non-parametrized model computationally tractable, it is common to restrict the structure of the preference rankings.

1.2 Overview of Choice Models

We give an overview of the popular choice models in the literature. This section is intended to give an understanding of the models that have been studied and the progression of the assortment optimization literature. This section also provides background knowledge to understand the importance and relevance of the new problems that we introduce in Section 1.3.

For cleaner presentation, we can also refer to the assortment using vector notation. Given an assortment S , we construct $\mathbf{x} \in \{0, 1\}^n$ such that $x_i = \mathbb{1}[i \in S]$. We abuse terminology slightly and refer to both S and \mathbf{x} as the assortment offered by the retailer.

1.2.1 Multinomial Logit Model

One of the most well-known choice model is the multinomial logit model (MNL). MNL falls into the class of random utility models (RUM), where a customer associates utility $\mu_i + \epsilon_i$ for each product i and chooses the product that gives her the highest utility. The first part of the summation is the mean utility for product i , μ_i , and the second part is the random noise in utility, ϵ_i . Customers always have access to the no-purchase option, which represents her ability to leave the store empty-handed. We denote the no-purchase option as product 0, with mean utility μ_0 and random noise in utility ϵ_0 .

What sets MNL apart from other models in the class of RUM is how ϵ is distributed. In MNL, we require that ϵ_i are independent and identically distributed according to the Gumbel distribution with mean zero and scale parameter one,

for $i \in N \cup \{0\}$. Due to the distribution on ϵ_i , the probability a customer has highest utility for product i and purchases this product as a result is:

$$P_i(\mathbf{x}) = \frac{e^{\mu_i} x_i}{e^{\mu_0} + \sum_{j=1}^n e^{\mu_j} x_j}.$$

Notice that the customer has non-zero probability of purchasing product i only when product i is offered in the assortment ($x_i = 1$). It is common to write $v_i = e^{\mu_i}$ and call v_i the preference weight of product i . Then the probability that a customer purchases product i is proportional to her preference weight for product i within the assortment \mathbf{x} , plus the no-purchase option.

MNL displays the Independence of Irrelevant Alternatives (IIA) property (Luce (1959), McFadden (1973)). The IIA property states that the ratio of purchase probabilities between two products, i and j , remains the same regardless of what other products are being offered. Mathematically, for assortments S_1 and S_2 that contain both products i and j , we have $P_i(S_1)/P_j(S_1) = P_i(S_2)/P_j(S_2)$. On one hand, it seems rational that offering more alternatives should not affect how a customer compares two products. On the other hand, the IIA property implies that the cannibalization effect from a new product depends solely on the preference weights of the original products in the assortment. As a result, all of the original products lose the same fraction of demand when a new product is introduced. This is problematic when the new product is a closer substitute for one of the original products, because we expect more cannibalization amongst similar products. Ben-Akiva and Lerman (1985) demonstrated the issue via the red bus-blue bus paradox. Suppose a transportation company can offer three services: transportation by taxi, red bus, and blue bus. Further suppose that when the assortment is $S = \{\text{taxi, red bus}\}$, we have $P_{\text{taxi}}(S) = P_{\text{red bus}}(S)$. Then the IIA property tells us that $P_{\text{taxi}}(S \cup \{\text{blue bus}\})/P_{\text{red bus}}(S \cup \{\text{blue bus}\}) = P_{\text{taxi}}(S)/P_{\text{red bus}}(S) = 1$. This implies that

$P_{\text{taxi}}(S \cup \{\text{blue bus}\}) = P_{\text{red bus}}(S \cup \{\text{blue bus}\})$, so that the blue bus is an equal substitute to the other two modes of transportation. This does not make practical sense, as the blue bus is a better substitute of the red bus. We should expect more customers to substitute from the red bus to the blue bus than from the taxi to the blue bus. The IIA property does not recognize that some products are closer substitute than others.

The optimal assortment of Problem (1.1) under MNL is always revenue-ordered (Talluri and Van Ryzin (2004), Liu and Van Ryzin (2008)). This means that if the products are sorted by their revenues so that $\pi_1 \geq \pi_2 \geq \dots \geq \pi_n$, then the optimal assortment takes the form of $\{1, \dots, k\}$ for some $1 \leq k \leq n$. One efficient algorithm for computing the optimal assortment would be to sort the products and compare the expected revenues from the n possible revenue-ordered assortments.

Due to the simplicity of MNL, the literature has studied the assortment optimization problem with various constraints and customer dynamics. Rusmevichientong et al. (2010) studied assortment optimization under MNL with a cardinality constraint, where the retailer chooses an assortment to maximize his expected revenue, subject to a limit B on the number of products that he can offer: $\sum_{i=1}^n x_i \leq B$. An optimal assortment which satisfies the cardinality constraint can be computed in polynomial time. However, assortment optimization under MNL with a shelf-space constraint is NP-hard (Rusmevichientong et al. (2009)). Under the shelf-space constraint, each product requires $b_i \geq 0$ units of space on the shelves. The assortment cannot use more shelf-space than a given budget B , such that $\sum_{i=1}^n b_i x_i \leq B$. Such constraints are useful when we consider a retailer operating a physical store with limited floor space. These constraints can also be

useful in an online setting, because there is limited space on a computer screen and customers tend to focus on products placed higher on a webpage.

MNL has also been used as the underlying choice model to study assortment optimization problems where the retailer has limited resources to produce products for customers who arrive over time. In the network revenue management problem, for example, there are m resources and the retailer has access to c_j units of resource j . If a customer purchases product i , then product i consumes $a_{i,j}$ units of resource j for $j = 1, \dots, m$. For example, an airline company sells an assortment of departure-destination routes, and each route requires the customer to travel and transfer on several flight legs. Customers arrive over a finite selling horizon. The airline chooses an assortment for each customer to maximize his total expected revenue from all customers, subject to the number of seats available on each flight leg. There is a tradeoff between reserving the seats on popular flight legs for routes with higher profits, and ensuring that the seats are sold before the departure date. Talluri and Van Ryzin (2004) studied the special case where every product consumes one unit of a universal resource using dynamic programming. The dynamic program is tractable and they showed that it is always optimal to offer revenue-ordered assortments. Furthermore, the assortment grows larger if the retailer is approaching the end of the selling horizon, and the assortment shrinks if the retailer is running out of the universal resource. In the general setting, the state space for the dynamic program is too large and the literature approximates the problem with a deterministic linear program, which assumes the realized demand is equal to the expected demand. The related linear program is exponential-sized if constructed naïvely (Gallego et al. (2004)). Gallego et al. (2014) presented a polynomial-sized linear program which can recover the optimal solution of the original linear program.

1.2.2 Extensions to MNL

In addition to considering MNL under different constraints and customer dynamics, the literature studies assortment optimization under variations of the MNL. Such variations are meant to better explain customers' decision processes and to resolve the limitations of MNL.

MNL has the IIA property and a new product serves as an equal substitute to all of the offered products. This is not true in practice, where similar products are more likely to cannibalize each other's demand. For example, an electronic retailer who introduces a new laptop would see more demand shift from other laptops than he would see from televisions. The nested logit model resolves this issue by grouping products into nests of similar products, so that a new product cannibalizes a larger fraction of market shares from other products in its nest. In the above example, the nests could be laptops and televisions. A customer first chooses the nest of products that she is interested in, and then makes a purchase within the chosen nest. The IIA property can also be removed by considering multiple customer types, as in the mixed multinomial logit model (MMNL), where each customer type follows MNL with different preference weights.

In the nested logit model, suppose there are K nests and n products per nest. Let N_k denote the universe of products belonging to nest k . Let $v_{k,i}$ be the preference weight of the i -th product in nest k . We also associate a dissimilarity parameter $\gamma_k \in [0, 1]$ with each nest, such that larger values of γ_k means that products in the nest are less similar. Suppose the retailer offers assortments \mathbf{x}_k for products in N_k for $k = 1, \dots, K$. Then the probability that a customer selects

nest k , $Q_k(\mathbf{x}_1, \dots, \mathbf{x}_K)$, is:

$$Q_k(\mathbf{x}_1, \dots, \mathbf{x}_K) = \frac{\left(\sum_{j=1}^n v_{k,j} x_{k,j}\right)^{\gamma_k}}{v_0 + \sum_{\ell=1}^K \left(\sum_{j=1}^n v_{\ell,j} x_{\ell,j}\right)^{\gamma_\ell}}.$$

If we fix assortments $\mathbf{x}_{k'}$ for all $k' \neq k$, then the retailer can increase the probability that a customer purchases from nest k by offering more products in nest k . In standard nested logit, a customer then chooses a product within assortment \mathbf{x}_k according to MNL, conditional on the customer selecting nest k above. The no-purchase option is not available once she has selected nest k . By constraining $\gamma_k \in [0, 1]$, the nested logit model falls into the class of RUM such that the random noises are correlated within nests (McFadden (1980)). As a result, the IIA property holds amongst products within nests but not across nests. Under these conditions, Davis et al. (2014) and Li and Rusmevichientong (2014) showed that the assortment optimization problem under the nested logit model is solvable in polynomial time. When γ_k is allowed to be greater than 1, the nested logit model can incorporate synergy effect amongst products. Synergy is the opposite of cannibalization, so that certain products look more attractive in a larger assortment. Davis et al. (2014) showed that assortment optimization is NP-hard when the dissimilarity parameter is allowed to be greater than 1.

MMNL resolves two issues with MNL. Other than having the IIA property, another concern with MNL is that all customers have the same preferences over products. In MMNL, there are L types of customers, so that each type has a different vector of mean utilities for the products and hence different preference weights (McFadden and Train (2000), Bront et al. (2009)). Let \mathbf{v}_ℓ be the preference weights of type $\ell = 1, \dots, L$, with preference weight $v_{\ell,0}$ for the no-purchase option. Furthermore, with probability λ_ℓ , the arriving customer belongs to type ℓ . All customers purchase from the same assortment \mathbf{x} . The purchase probability

of product i under MMNL is:

$$P_i(\mathbf{x}) = \sum_{\ell=1}^L \lambda_{\ell} \cdot \frac{v_{\ell,i} x_i}{v_{\ell,0} + \sum_{j=1}^n v_{\ell,j} x_j}.$$

McFadden and Train (2000) showed that MMNL also belongs to the class of RUM by appropriately defining the distribution of the random noises. They also showed that MMNL can approximate any model in the class of RUM. Rusmevichientong et al. (2014) showed that the assortment optimization problem is NP-hard, even when there are only two types of customers. Furthermore, the performance of the revenue-ordered assortment can be arbitrarily bad as the number of products or customer types grows. Désir et al. (2014) presented a FPTAS for assortment optimization problem under MMNL, which also works when we have a cardinality or shelf-space constraint.

Many other choice models have been defined by changing the distributions and correlation structures of the random noise ϵ . Examples include the generalized logit model which allows products to be in multiple nests, and the probit model where ϵ follows a multivariate normal distribution. Details can be found in the work of Train (2003).

1.2.3 Non-Parametric Choice Models

Another way to represent customer preferences is to use ranked lists. Of all the products in N , a customer is interested in only a subset of products, which we call her consideration set. A customer's consideration set, $C \subseteq N$, is the set of products that she is willing to purchase. Her consideration set is independent of the assortment being offered by the retailer. For example, a customer who wants to purchase a new television would have the set of all televisions in N

as her consideration set, and she would not purchase any product other than a television. The customer either purchases a product in $C \cap S$ or leaves without a purchase. She ranks the products in her consideration set C in order of decreasing preferences. This ranking over C is called her preference list, which we denote by σ . We use the notation $i \succ_{\sigma} j$ to denote that product i is preferred over product j in the preference list σ .

Since a customer only ranks the products in her consideration set, we say that a product is in her consideration set or in her preference list interchangeably. A customer purchases the highest-ranked product in her preference list which is available in the assortment. If none of the products in her preference list are available, then she leaves the store without a purchase.

Let Σ be the set of all preference list σ . With probability λ_{σ} , the arriving customer has preference list σ . The purchase probability of product i is:

$$P_i(\mathbf{x}) = \sum_{\sigma \in \Sigma} \lambda_{\sigma} \left(x_i \cdot \prod_{j: j \succ_{\sigma} i} (1 - x_j) \right).$$

In the above computation, we recognize that a customer who has preference list σ purchases product i if and only if product i is available ($x_i = 1$) and none of the products which precede product i are available ($x_j = 0$ for all $j \succ_{\sigma} i$). This non-parametrized choice model is not very useful computationally because there are $O(n!)$ possible preference lists. Hence, it is common to restrict the structure of the preference lists in our models. Paul et al. (2016), Feldman et al. (2017), Aouad, Farias and Levi (2015) focused on restricting the size of the consideration sets, and hence the length of σ . This captures the idea that customers are only interested in a small number of products in the store. Goyal et al. (2016), Aouad, Farias, Levi and Segev (2015), Aouad, Farias and Levi (2015) studied preference lists which form nests or intervals over a central, objective ranking. These struc-

tures represent the cases where products can be objectively ranked, and each customer has one or more conditions for the products that she is willing to purchase. For example, a retailer offers several products which increase in quality as price increase. If customers are price-conscious, the central ranking would order the products by increasing price. Each customer has a minimum quality requirement for her purchase and a maximum budget that she can spend. Hence, her preference list would be a interval on the central ranking.

1.3 Future of Assortment Optimization

Given the abundance of literature in assortment optimization, what should be the direction of future research in order to maximize its relevance to industry? Feldman et al. (2018) answered this question in a recent field study in collaboration with online retailer Alibaba. Previously, Alibaba relied on machine learning algorithms to predict the probabilities that customers purchase products. The machine learning algorithms assumed that a customer's purchasing decision is independent of the products that are included in the assortment, so that $P_i(S) = p_i$ for all $S \subseteq N$. In other words, these techniques do not consider substitution effects when more than one product is shown to the customer. One reason that standard machine learning techniques cannot consider substitution effects is that the retailer would need to predict a purchase probability for each product under each assortment, $P_i(S)$, and there are an exponential number of terms to estimate and compare. Instead, Feldman et al. (2018) used the maximum likelihood estimator to estimate the parameters of MNL for each customer, and then offered a MNL-optimal assortment. They observed an increase in revenue 28% per customer visit. They argued that retailers should incorporate assort-

ment optimization techniques into the popular machine learning algorithms to improve revenue. Feldman et al. (2018) suggested one future research direction is to study choice models that reflect the current retail environment more accurately, which is the purpose of this thesis.

In Subsections 1.3.1-1.3.3, we present three new assortment optimization problems which we study in this thesis. We identify observations in the recent marketing literature which have not been adequately captured by the traditional choice models in Section 1.2, and study the corresponding assortment optimization problem.

1.3.1 Omnichannel Assortment Optimization

Today, many retailers operate both a physical store and an online store. It is common for customers to visit one or both stores, also known as channels, before purchasing from either of the stores. This phenomenon has been well-studied in the marketing literature (Bell et al. (2017), Fornari et al. (2016), Avery et al. (2012), Bell et al. (2014)). For example, a customer can visit a physical store to try out a red skirt, but end up purchasing online because she wants the same skirt in black and it is not sold in-store. Visiting the store allows her to determine her size and feel the material. A retailer should make operating decisions for both selling channels as one cohesive unit, because he knows that the customer will utilize both channels to make her purchasing decision. This is referred to as the omnichannel retail environment and we refer to the retailer as an omnichannel retailer. The models in Section 1.2 are insufficient because they assume that a customer observes and purchases from the same assortment. In

omnichannel retailing, a customer can observe multiple assortments from the same retailer before purchasing from her channel of choice.

We focus on the assortment optimization problem faced by an omnichannel retailer in Chapter 2. The retailer offers the full assortment N in the online store, and a subset S in the physical store. There are two types of customers, offline customers and online customers. The customer's type refers to the store which she purchases from. An offline customer visits the physical, offline store to purchase directly from this store. Using MNL as the underlying choice model, an offline customer either purchases from the in-store assortment S , or leaves without a purchase. An online customer visits the physical store to observe the products in S before purchasing from the online store. An online customer purchases from the full, online assortment N . Her purchasing decision also follows MNL, but her preference weights depend on the products that she observes in-store. Following the structure of MMNL, an online customer arrives at the retailer with probability λ , and an offline customer arrives with probability $1 - \lambda$.

Products consist of features, and a feature can be shared amongst several products. To describe how features are shared amongst products, we use a features tree, T , where the leaf vertices represent the products and the non-leaf vertices represent features. The path from a leaf to the root describes the features of the product located at the leaf. The features shared by two products are represented by their common ancestors on T . A feature is observed by an online customer if any of the products that share this feature is offered in the in-store assortment S . Each feature is associated with a boost or discount parameter that describes whether the feature is well-liked or disliked by an average online customer, if she has the opportunity to observe the feature. Her preference weights

for the products in N is a function of the features that she observes when she visits the physical store.

The goal is to choose an assortment to offer in the physical store which maximizes the retailer’s total expected revenue from both types of customers. Similar to the assortment optimization problem under MMNL, we show that the omnichannel assortment optimization problem is NP-hard when there are both offline and online customers ($\lambda \in (0, 1)$). We begin by studying the case with only online customers ($\lambda = 1$), so that the in-store assortment is simply a display assortment. We present a polynomial-time algorithm which computes the optimal assortment to display in the physical store. We extend our algorithm into a FPTAS when there are both offline and online customers. The runtime of our FPTAS is polynomial in the number of products, the input parameters, and the desired accuracy.

Finally, we also test the modeling capability of our features tree model and the practical performance of our FPTAS. In particular, we consider products which can share features arbitrarily and approximate this generalized model with a features tree. Our features tree model sacrifices a small percentage of expected revenue to gain computational tractability. Our FPTAS achieves a much larger fraction of expected revenue than our theoretical guarantee on average.

1.3.2 Assortment Optimization in the Presence of Synergy

One common trait of the models in Section 1.2 is that a new product must cannibalize the market share of the original products. Demand for products, which is represented by purchase probabilities, cannot increase by enlarging the as-

sortment. Although this seems to make sense when we consider customers who are rational and make utility-maximizing decisions, the marketing literature observes an opposite effect as well. By introducing certain products to the assortment, a retailer can increase demand for some other target products. In one study, a retailer selling a bread maker introduced a second, over-priced alternative to his store (Simonson (1999)). Customers perceived more value for the initial bread maker and demand increased. When a product can increase another product's perceived value by being in the assortment, we say that the two products create synergy.

We consider a choice model introduced by Feng et al. (2015) which incorporates synergy effects explicitly. In Chapter 3, we study an assortment optimization problem with synergy effects. We consider a retailer with access to the products in N . In addition to the base preference weights for products, each pair of products have a pair of synergy weights. When products i and j are both included in the assortment S , then the preference weights of the two products are increased by the appropriate synergy weights. This can be interpreted as a perceived increase in utility when products are observed side-by-side, even when the customer only intends to purchase one of the products. We use MNL as the underlying choice model, except that the preference weights are functions of the assortment being offered. The goal is to choose an assortment which maximizes the retailer's expected revenue.

The assortment optimization problem where arbitrary pairs of products create synergy is NP-hard, and is related to maximizing a quadratic function with binary variables. Instead, we focus on synergy effects between specific pairs of products. To describe the synergy structure, we create a synergy graph where

each product is represented by a vertex. There is an edge between two products if they create synergy when they are both included in the assortment. When the synergy graph is a path or a tree, we present efficient algorithms to compute the optimal assortment via dynamic programming. We extend our algorithm for the synergy tree to consider synergy graphs with low treewidth. When the synergy graph is a path, we also present a linear program which can recover the optimal assortment.

1.3.3 Assortment Optimization with Dynamic Substitution and Stocking Costs

In Chapter 4, we study an assortment optimization problem where customers arrive over time. Unlike the network revenue management problem discussed in Section 1.2, we do not allow the retailer to adjust the assortment seen by each customer. Rather, the retailer decides on an assortment before customers arrive, as well as stocking levels for the products in the assortment. A customer makes a purchasing decision based on the products that are still in-stock upon her arrival. The assortment seen by each customer changes over time, depending on the purchases of the customers that arrived before her. This is referred to as dynamic substitution, because the customer's purchasing decision and substitution behaviour changes based on the assortment she encounters upon arrival.

Dynamic substitution behaviour can be seen in both physical stores and online stores. In a grocery store, the retailer cannot restock or remove products before the arrival of each customer. Even in online stores, major retailers like Apple and Gap offer the same assortment to every customer until products run

out. Hence, the assortment optimization problem with dynamic substitution has many applications.

Aouad, Levi and Segev (2015*a,b*), and Goyal et al. (2016) all consider assortment optimization with dynamic substitution, such that there is a constraint on the total units of stocked inventory. Instead, we consider the problem with a stocking cost per unit of product. The goal is to maximize expected profit instead of expected revenue.

Customers arrive one-by-one into the store to make a purchase. We use the non-parametric preference ranking choice model as the underlying choice model. Each customer purchases the first product in her preference list which is still in-stock when she arrives at the store, and leaves the store if all the products in her preference list have stocked-out. We refer to the aggregate information on the number of customers arriving to the store, their preference lists, and the order in which they arrive as the customers arrival sequence. In the stochastic setting of the assortment optimization problem, the customers arrival sequence is random and drawn from some finite probability distribution. The problem is to select an assortment with stocking levels which maximizes the retailer's expected profit.

The stochastic setting is extremely difficult, and we begin by studying the deterministic setting of our problem. In the deterministic setting, the customers arrival sequence is revealed, so that we know the preference lists of all the customers before we decide on an assortment with stocking levels. Our goal is to maximize the retailer's profit. We construct efficient algorithms to compute either an optimal or approximately-optimal assortment with stocking levels for the deterministic setting, depending on the structure of the preference lists dis-

cussed below. We use our algorithms from the deterministic setting to create a heuristic for the stochastic setting.

We show that even when customers consider only two products, the deterministic setting is still NP-hard. To contrast with the result of Paul et al. (2016), they showed NP-hardness when there is one customer of unknown preferences arriving into the store, whereas we showed NP-hardness when there are many customers for whom we know their preference lists. We transform the problem so that the retailer decides on an initial assortment and product discontinuation times rather than the stocking levels. The transformation simplifies the analysis when we restrict the structure of the preference lists. When customers consider at most K products, we present an algorithm which guarantees $\frac{2}{K} \cdot (1 - K)^{K-1}$ -fraction of the optimal profit. We also study the case where customers' preference lists are intervals on a central ranking on the products. In this case, we present an efficient algorithm which computes the optimal assortment with stocking levels. The runtime of our algorithm is polynomial in the number of products and customers when the maximum interval length is fixed, and exponential in the maximum interval length when the intervals can be arbitrarily long.

We use our algorithm in the deterministic setting to construct a heuristic for the stochastic setting. Our heuristic samples the customers arrival sequences and computes an assortment with stocking levels for each realized outcome. It chooses the assortment and stocking levels which achieves the highest profit on average over the sample set. We find that our heuristic performs reasonably despite its simplicity.

1.4 Organization and notation

In Chapter 2, we consider assortment optimization for an omnichannel retailer. We introduce the problem and provide a literature review in Section 2.1. In Section 2.2, we describe the omnichannel assortment optimization problem under the features tree model. We describe how products are related by features on the features tree and how an online customer would update her preferences for the online products based on the in-store assortment. In Section 2.3, we focus on the case where all customers are online customers, and we give a polynomial-time algorithm to compute the optimal assortment to display in the physical store. In Section 2.4, we consider the problem with both online and offline customers, and present a FPTAS to compute an assortment which achieves $(1 - \epsilon)$ -fraction of the optimal expected revenue. Here, $\epsilon > 0$ is an accuracy parameter which we may control. We present extensions in Section 2.6, where the retailer can choose both the online and physical stores' assortments. Finally, we provide numerical experiments in Sections 2.7 and 2.8. We test the modeling power of our features tree model when the ground-truth model allows products to have arbitrary sets of features. We also assess the practical performance of our FPTAS. Some proofs and details are deferred to Appendix A.

In Chapter 3, we consider assortment optimization with synergy effects between products. We introduce the problem and provide a literature review in Section 3.1. In Section 3.2, we describe synergistic MNL and the synergy graph. In Section 3.3, we focus on the cases where the synergy graph takes the form of a path or a tree, and use dynamic programming to compute the optimal assortment in each case. We also consider the assortment optimization problem for synergy graphs with low treewidth. In Section 3.4, we present a linear program

which recovers the optimal assortment when the synergy graph is a path. Some proofs and details are deferred to Appendix B.

In Chapter 4, we consider assortment optimization with dynamic substitution and inventory stocking costs. We introduce the problem and provide a literature review in Section 4.1. In Section 4.2, we present our model and notation. In Section 4.3, we show that the problem of finding the optimal assortment with stocking levels is NP-hard, even when the customers arrival sequence is revealed to the retailer. We formulate an integer program for the deterministic setting. We transform the problem, so that the retailer decides on the optimal time periods to discontinue products rather than the stocking levels. We proceed to study the problem when customers are K -choosy and when customers have preference lists which are intervals on a central ranking. In Section 4.4, we present our heuristic for the stochastic setting, and a linear program to upper-bound the average profit over a sample set of customers arrival sequences. We test our approximation algorithm for K -choosy customers and the heuristic for the stochastic setting in Section 4.5. Some proofs are deferred to Appendix C.

Variables and notations defined are usually specific to the chapter. In general, a bold font \mathbf{x} represents a vector. For example, if \mathbf{x} is the vector representing an assortment of products, then $\mathbf{x} = (x_i : i \in N)$.

CHAPTER 2

OMNICHANNEL ASSORTMENT OPTIMIZATION UNDER THE MULTINOMIAL LOGIT MODEL WITH A FEATURES TREE

2.1 Introduction

Historically, retailers have operated on a single channel, either as pure offline brick-and-mortar stores or pure online stores. As online shopping has become ubiquitous, customers no longer use a single channel to both research and purchase products (Bachrach et al. (2016)). A customer can test out products at a local retailer before deciding to purchase online, a practice known as showrooming. In response, retailers have shifted away from operating a single channel to selling on multiple channels. Traditional brick-and-mortar stores like Best Buy and Wal-Mart operate websites and offer products that may not be available in-store. Diamonds retailer Blue Nile and eyeglasses retailer Warby Parker started as online retailers, but have opened showrooms to display their products. Recent literature refer to this phenomenon as an omnichannel retail environment because retailers must operate multiple channels as one cohesive unit. In this environment, products may share features so that if a retailer displays a product in-store, the customer can try out the product and modify her preferences of other online products based on similar features. This leads to the study of assortment optimization from an omnichannel viewpoint.

In our paper, we study an assortment optimization model for an omnichannel retailer operating a physical store and an online store. The retailer has n products at his disposal, and offers the full assortment of n products in his online store. He selects a subset of the full assortment for his physical store. Each

product consists of several features, and a feature can be shared amongst several products. There are two types of customers: offline and online. An offline customer visits the physical store to purchase from the assortment she sees in-store. An online customer visits the physical store to test out the products before purchasing from the full assortment online. By trying out the products in-store, she learns whether the products' features are over- or under-valued, and can update her preferences of online-only products based on the features that are shared with the displayed products. She does not have to observe an identical product in-store before updating her preferences online. For example, a customer purchasing a phone would update her opinion of a silver iPhone 8 if she sees a gold iPhone 8, because the phones share many features and experiencing the common features on one phone changes her opinion on the other phone.

The physical store serves as a display front for online customers to test out the products and update product preferences on a feature-by-feature basis, and as the only point of sales to offline customers. The assortment optimization problem is to select a subset of the products from the online assortment to display in the physical store to maximize the expected revenue.

Our Contributions

We consider a retailer operating two channels: an online store and a physical store. The retailer offers the full assortment online and a subset in his physical store. Each product is associated with a revenue, and the probability that a customer purchases product i depends on her preference weight of product i and her purchasing channel. Products have features and we describe simi-

larities amongst products by their shared features. We model product features with a tree (see Figure 2.1). Each vertex of the tree is a feature and each leaf corresponds to a product, such that the path from a leaf to the root gives all the features that uniquely defines a product. Two products share a feature if that feature is a common ancestor on the tree.

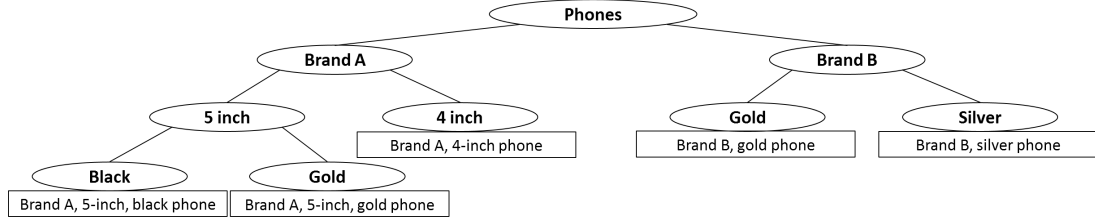


Figure 2.1: Features tree of five phones. Ovals describe features and boxes are the products that result from a set of features. There is one 4-inch phone under Brand A, so we do not differentiate by colours. There is no size feature for a Brand B phone.

Offline and online customers purchase according to the multinomial logit model (MNL). An offline customer decides amongst the products in the in-store assortment; her preference weights for products are given as input parameters and fixed. An online customer visits the physical store to test out features before purchasing from the full assortment online. She updates her preference weights in the online store using the features tree.

The goal is to choose an assortment to display in the physical store which maximizes the retailer's expected revenue across offline and online customers. A mathematical representation will be presented in the next section. We call this problem the OmniChannel Assortment optimization problem (OCA) under a features tree model.

We show that the OCA problem under a features tree model with offline and online customers is NP-hard via a reduction from the partition problem. We

refer to the problem with both types of customer as the “general” setting. This is common amongst physical retailers that started their own online store (e.g. Best Buy). The retailer uses the physical store as a display front and to target traditional or impatient customers who consider only the products available at the store.

Since the general setting is NP-hard, we begin by studying the special case of the “showroom” setting, where all customers are online customers and the physical store is a display front. This is common amongst historically online retailers which later opened their own showrooms (e.g. Blue Nile). A customer sees a subset of products in-store and can test out different sizes and settings, but she can only order her ring online with all the required modifications. We present an algorithm that finds the optimal display assortment with runtime polynomial in the number of products.

We present a fully polynomial time approximation scheme (FPTAS) for the general setting, so that its runtime is polynomial in the number of products, the input parameters, and the desired accuracy. Our FPTAS involves creating a geometric grid over the numerator and denominator of the expected revenue function from offline customers. For each point in our grid, we construct a fractional combinatorial optimization problem which requires us to optimize the expected revenue of the online customers subject to constraints defined by the grid point. We parametrize our problem by the online expected revenue, and present a dynamic program to solve our parametrized problem. We compute an assortment for every point in the grid and choose the assortment with the largest total expected revenue amongst all points in our grid.

Literature Review

To the best of our knowledge, Dzyabura and Jagabathula (2017) were the first to study the assortment optimization problem of an omnichannel retailer. They considered feature classes with several feature values per class. A product consists of one feature value per feature class. In Figure 1, the feature classes are the brand, size, and colour, and the feature values in colour are gold, silver and black. The novelty is that they optimized over the sets of features to display and recovered the optimal assortment from the features set. In the showroom setting, their model is solvable in polynomial-time only when products have unit-revenue, and they presented a FPTAS for arbitrary revenues. In the general setting, it is difficult to recover an optimal assortment; various assortments can display the same set of features but earn different offline expected revenue. They restricted the space of feasible assortments to only the maximum assortments described by sets of features, and gave a FPTAS when the size of the feature classes is fixed. Finally, they demonstrated empirical evidence that customers update preferences by features via a field experiment; participants rank various messenger bags before and after observing a subset of similar bags.

Dzyabura and Jagabathula (2017) allowed products to share feature values over any feature classes, but we restrict features and products to a tree. In Figure 2.1, an online customer in their model would update her preference for colour on both Brand A and B gold phones when either are displayed. In our model, her preference for a Brand B gold phone with regards to colour is unaffected when she sees a Brand A gold phone. Colour is not a feature shared across brands, and seeing the shade of gold from the former brand does not give her information about the latter. Their model is able to describe product relation-

ships more generally. On the other hand, they required that a product exists for every combination of feature values, so that the online store must carry phones in two sizes for both brands, whereas we recognize that some brands do not offer size options. Our model is more flexible with respect to the products that must be carried by the retailer in the online store. Hence neither model can be considered a generalization of the other and each has its strength. Finally, the features tree model is not NP-hard in the showroom setting, and allows us to optimize over all assortments in the general setting without restricting the size of feature classes. Trees have been used to describe product taxonomy in market segmentation and recommender system construction (Albadvi and Shahbazi (2009), Cho et al. (2002), Gangurde and Akarte (2015), Ziegler et al. (2004)).

Other works have studied challenges in the omnichannel environment. Balakrishnan et al. (2014) modeled the competition between a physical retailer and an online retailer, where customers may visit the physical retailer to test products before purchasing from the online retailer. Gao and Su (2016b) considered an omnichannel retailer who sells one product and shares information with customers to increase profit. To provide information about the online store, a display model is available in-store so that customers can try out the product and purchase online, even if it is sold out. To provide information on the physical store, in-store availability is posted online so that customers can evaluate the utility of visiting the physical store. They classified regimes where each option improves profit. In a similar paper, Gao and Su (2016a) considered an omnichannel retailer with a Buy-Online-Pickup-Instore (BOP) option, where customers can order before pick up to mitigate stock-out risk. This model incorporated the store’s ability to cross-sell when customers visit, and they gave conditions un-

der which BOP increased profitability. These papers considered a single product with independent demand, but we integrate a choice model so that demand depends on the assortment. They assumed that each customer has a private valuation which is revealed after seeing the product, whereas we assume a shift in market preference when features are seen.

Empirical evidence supports the importance of studying an omnichannel retail environment. Bell et al. (2017) investigated glasses retailer Warby Parker, which used to sell online exclusively and had a sampling program for customers to try glasses for five days. They found that online sales increased and returns decreased in cities where Warby Parker opened a showroom to let customers visit and try out frames before ordering online. Fornari et al. (2016) and Avery et al. (2012) looked at adding a physical store to the online store for an electronics retailer and a high-end apparel and furnishing retailer respectively. Both studies found that online sales increased in the long-run when customers have an additional channel to research products. In the reverse direction, customers like to purchase in-store, especially when products are “high-touch” (e.g. clothing, furniture) (Bell et al. (2014)). Retailers benefit from the opportunity to cross-sell products and Bell et al. (2014) found that BOP increases store visits and revenue.

Our work is also related to the large body of literature on assortment optimization. Our underlying choice model is MNL, which is credited to Luce (1959) and McFadden (1973), with the additional interpretation that a product’s mean utility depends on whether its features are observed or not. A similar tree taxonomy was used by Li et al. (2015) in their d-level nested logit model to describe product features, where each vertex represents a feature that is shared amongst leaves in its subtree. The important difference is that their tree de-

scribed a customer's choice process whereas our tree is used to update product preferences. In Li et al. (2015), a customer decides on the feature she likes and shrinks the assortment from which she is willing to purchase from as she moves from the root to a leaf. In our model, a customer always consider the entire assortment available either online or offline, depending on her type.

Finally, the existence of offline and online customers is related to the mixture of MNL (MMNL) studied by Bront et al. (2009) and Rusmevichientong et al. (2014). In MMNL, multiple customer types consider the same assortment, but each type has different product preferences. In our model, all customers can have the same preferences, but consider different assortments based on their purchasing channel. Our FPTAS uses techniques from Désir et al. (2014)'s paper on capacitated MMNL, where each product has a capacity requirement and the assortment's capacity cannot exceed a budget. For each customer type, they created a geometric grid on the numerator and denominator of the expected revenue function. A point on the grid lower-bounds the expected revenue to be earned from each customer type. They gave a dynamic program that finds the minimum capacity assortment which satisfies the constraints imposed by the grid, if such an assortment exists. Our FPTAS creates a similar geometric grid on the numerator and denominator of the offline expected revenue, and we maximize the online expected revenue.

Organization

In Section 2.2, we describe the OCA problem under the features tree model. In particular, we describe how products are related by features on the features tree and how an online customer would update her product preferences based on

the in-store assortment. In Section 2.3, we focus on the showroom setting with only online customers and we give a polynomial-time algorithm. In Section 2.4, we present the FPTAS for the general setting. In order to consider the practical performance of our FPTAS, we also provide an efficient method to upper-bound the optimal expected revenue in Section 2.5. In Section 2.6, we present extensions where the retailer can choose both the online and physical stores' assortments.

We provide numerical experiments in our last two sections. In Section 2.7, we test the modeling power of our features tree model when the underlying ground-truth model allows for more general features relationship across products. In Section 2.8, we assess the practical performance of our FPTAS against the upper-bound described in Section 2.5. Finally, we conclude in Section 2.9.

2.2 The Model

We consider a retailer operating an online store and a physical (offline) store. There are n products in the online store, denoted $N = \{1, \dots, n\}$. Product i generates a revenue of π_i when it is purchased by a customer. All products are offered online and the retailer's decision is to select an assortment $S \subseteq N$ for the physical store, which we call the display assortment or the in-store assortment.

Products have features, and we use a features tree T to describe how features are shared amongst products. The vertices of the tree correspond to features of products, and the leaves correspond to products. The feature at a leaf is the final feature differentiating the corresponding product from the other children of its parent. The path from a leaf to the root gives all the features that uniquely

defines the product. See Figure 2.1 for an example.

To describe the structure of the tree, we introduce notations to describe its parent-child relationships. We assume without loss of generality that our features tree T is a binary tree.

Assumption 1. *The features tree T is a binary tree so that every non-leaf vertex has exactly two children. A binary tree T with n leaves always has $2n - 1$ vertices, hence we index the leaves by $N = \{1, \dots, n\}$ and non-leaves by $\{n + 1, \dots, 2n - 1\}$.*

If k is not the root, then we denote $p(k)$ as the parent of k in T . If k is not a leaf, then it has two children; we denote the left child by $\ell(k)$ and the right child by $r(k)$. For all vertices k , let $A(k)$ be the set of ancestors of vertex k and itself. Let $L(k)$ be the set of leaves in the subtree rooted at k . Then $L(k)$ is the set of products which share feature k . Recursively, these are defined as:

$$A(k) = \begin{cases} \{\text{root}\} & : k = \text{root} \\ A(p(k)) \cup \{k\} & : k \neq \text{root} \end{cases},$$

$$L(k) = \begin{cases} \{k\} & : k \in N \\ L(\ell(k)) \cup L(r(k)) & : k \notin N \end{cases}.$$

A feature k is observed if any product in $L(k)$ is made available in-store. When we offer assortment S in-store, the set of features demonstrated to customers is $\cup_{j \in S} A(j)$, and we say that feature $k \in \cup_{j \in S} A(j)$ is demonstrated in-store.

There are two types of customers in the market: offline and online customers. A customer's type determines the assortment she is purchasing from and her preferences, which are described by her preference weights for the products. An offline customer purchases only from the in-store assortment, and always

associates a preference weight of \hat{v}_i with product i . An online customer visits the physical store to observe the displayed products, but ultimately purchases from the full assortment N online. She updates her preferences based on the features of products in S and associates preference weight of $v_i(S)$ with product i . We first describe the process in which an online customer updates her preference weights, then define $v_i(S)$.

An online customer has initial preference weight w_i for product i , which is her preference weight when she does not have the opportunity to examine any features of product i . She also associates a boost or discount multiplier δ_k to vertex k in T , which represents her change in preference when feature k is observed. The customer updates her preference weight by multiplying w_i by δ_k if she sees feature k and k is a feature of product i . The reason behind using multiplicative updates will be clear when we justify our model in light of MNL later. A multiplier $\delta_k > 1$ corresponds to a feature being more appealing than presumed (undervalued), $\delta_k < 1$ corresponds to a feature being less attractive than presumed (overvalued), and $\delta_k = 1$ corresponds to no changes in opinion.

To define $v_i(S)$, we introduce binary variables x_k for every vertex k in the features tree, such that $x_k = 1$ means that feature k is demonstrated in-store and $x_k = 0$ otherwise. The latter case is equivalent to saying that none of the products having feature k is available in-store. Given an in-store assortment S , the corresponding characteristic vector $\mathbf{x} \in \{0, 1\}^{2n-1}$ is:

$$x_k = \begin{cases} 1 & : k \in (\cup_{j \in S} A(j)) \\ 0 & : k \notin (\cup_{j \in S} A(j)) \end{cases}.$$

If feature k is a leaf, then the retailer must offer product k when feature k is demonstrated in-store. Hence the first n indices of \mathbf{x} reveals the in-store as-

sortment. Given a vector \mathbf{x} , we can recover an assortment by taking the set of products such that $x_k = 1$ for $k \in N$.

If feature k is not a leaf, then it is demonstrated in-store if and only if at least one of the leaves in its subtree is available in-store. Such a leaf is in the subtree of either $\ell(k)$ or $r(k)$, and this means that at least one of features $\ell(k)$ or $r(k)$ is demonstrated. Hence $x_k = 1$ if and only if $x_{\ell(k)} = 1$ or $x_{r(k)} = 1$. We denote \mathcal{X} as the set of feasible characteristic vectors, such that:

$$\mathcal{X} = \left\{ \mathbf{x} \left| \begin{array}{ll} x_{\ell(k)} \leq x_k & \forall k \notin N, \\ x_{r(k)} \leq x_k & \forall k \notin N, \\ x_k \leq x_{\ell(k)} + x_{r(k)} & \forall k \notin N, \\ x_k \in \{0, 1\} & \forall k. \end{array} \right. \right\}.$$

Hence we can write the online preference weight of product i as $v_i(S) = v_i(\mathbf{x})$ for $\mathbf{x} \in \mathcal{X}$, where:

$$v_i(\mathbf{x}) = w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}.$$

An online customer either purchases a product in N or chooses the no-purchase option. An offline customer purchases exclusively from the in-store assortment \mathbf{x} or chooses the no-purchase option. The no-purchase option refers to the customer's ability to leave the store without making a purchase, and is available regardless of her shopping channel. This option does not share features with products, and its preference weight does not change regardless of the in-store assortment. We denote the preference weight of the no-purchase option by w_0 for an online customer and \hat{v}_0 for an offline customer.

A customer's purchase probability of product i is proportional to the preference weight of product i in the assortment which she purchases from, using the

structure in MNL. The purchase probability of product i for online customers, given the in-store assortment \mathbf{x} , is:

$$P_i^{ON}(\mathbf{x}) = \frac{v_i(\mathbf{x})}{w_0 + \sum_{j=1}^n v_j(\mathbf{x})} = \frac{w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}}{w_0 + \sum_{j=1}^n w_j \cdot \prod_{k \in A(j)} \delta_k^{x_k}}.$$

The purchase probability of product i for offline customers is:

$$P_i^{PHY}(\mathbf{x}) = \frac{\hat{v}_i x_i}{\hat{v}_0 + \sum_{j=1}^n \hat{v}_j x_j}.$$

We denote the expected revenue from online and offline customers by $\Pi^{ON}(\mathbf{x})$ and $\Pi^{PHY}(\mathbf{x})$ respectively, where

$$\Pi^{ON}(\mathbf{x}) = \sum_{i=1}^n \pi_i P_i^{ON}(\mathbf{x}), \quad \text{and} \quad \Pi^{PHY}(\mathbf{x}) = \sum_{i=1}^n \pi_i P_i^{PHY}(\mathbf{x}).$$

Let q be the fraction of online customers and $(1 - q)$ be the fraction of offline customers. When $q = 0$, then all customers are offline customers who purchase from the in-store assortment \mathbf{x} with preference weights \hat{v}_i for product i and this reverts back to MNL. Hence we consider $q \in (0, 1]$ in the OCA problem. If we display assortment \mathbf{x} , then we obtain an expected revenue of:

$$\Pi(\mathbf{x}) = q \cdot \Pi^{ON}(\mathbf{x}) + (1 - q) \cdot \Pi^{PHY}(\mathbf{x}).$$

The assortment optimization problem is to choose an assortment \mathbf{x} that maximizes the total expected revenue. The OCA problem can be formulated as the following optimization problem, where $\Pi^{ON}(\mathbf{x}), \Pi^{PHY}(\mathbf{x})$ has been expanded out:

$$\max_{\mathbf{x} \in \mathcal{X}} \Pi(\mathbf{x}) = \max_{\mathbf{x} \in \mathcal{X}} q \cdot \frac{\sum_{i=1}^n \pi_i w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}}{w_0 + \sum_{i=1}^n w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}} + (1 - q) \cdot \frac{\sum_{i=1}^n \pi_i \hat{v}_i x_i}{\hat{v}_0 + \sum_{i=1}^n \hat{v}_i x_i}.$$

To interpret our model as a utility-maximizing choice model, let $\mu_i(\mathbf{x})$ denote the utility of product i when assortment \mathbf{x} is displayed in-store. Following the setup of MNL, a customer's utility of product i is the mean utility of product i

plus a noise ϵ_i , which is generated by an independent, standard Gumbel random variable with mean 0. Before observing any features, online customers have a mean utility of $\ln w_i$ for product i . If feature k of product i is demonstrated in-store, then the mean utility of product i changes additively by $\ln \delta_k$, so that

$$\mu_i(\mathbf{x}) = \left(\ln w_i + \sum_{k \in A(i)} x_k \ln \delta_k \right) + \epsilon_i = \ln \left(w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k} \right) + \epsilon_i.$$

The additive updates to the mean utility of product i translate to multiplicative updates to the preference weight of product i .

2.3 Showroom Setting

We begin by considering $q = 1$, so that we are in the showroom setting with only online customers. Let γ^* be the optimal expected revenue:

$$\gamma^* = \max_{\mathbf{x} \in \mathcal{X}} \frac{\sum_{i=1}^n \pi_i w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}}{w_0 + \sum_{i=1}^n w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}}.$$

As in standard fractional combinatorial optimization theory, we can parametrize the objective function and find a fixed point to the parametrized problem. Specifically, suppose there exists an assortment $\mathbf{x} \in \mathcal{X}$ that generates expected revenue greater or equal to γ , so that

$$\frac{\sum_{i=1}^n \pi_i w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}}{w_0 + \sum_{i=1}^n w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}} \geq \gamma.$$

By rearranging this inequality, we observe that a display assortment \mathbf{x} generates expected revenue greater or equal to γ if and only if the same assortment satisfies $\sum_{i=1}^n (\pi_i - \gamma) w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k} \geq w_0 \gamma$. By maximizing the left side of this new inequality over all assortments in \mathcal{X} , we obtain our parametrized problem below.

Let $f(\gamma)$ denote the value of Problem (2.1) as a function of γ :

$$f(\gamma) = \max_{\mathbf{x} \in \mathcal{X}} \sum_{i=1}^n (\pi_i - \gamma) w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}. \quad (2.1)$$

Claim 1. *Given $f(\gamma)$ as defined above, let γ^* be the optimal expected revenue of the showroom setting. Then the following are true: i) $f(\gamma) > w_0 \gamma$ if $\gamma < \gamma^*$, ii) $f(\gamma) < w_0 \gamma$ if $\gamma > \gamma^*$, and iii) $f(\gamma) = w_0 \gamma$ if $\gamma = \gamma^*$.*

Finding γ^* requires a method to efficiently solve Problem (2.1) and a method to search for γ^* over possible values of γ . A feasible solution to Problem (2.1) satisfies exactly one of the following statements: either $\mathbf{x} = \vec{0}$ or there exists $i \in N$ such that $x_i = 1$, which is equivalent to $x_{\text{root}} = 1$ because the root is featured whenever a product is offered in-store. Our dynamic program considers non-empty assortments and compares the result to $\mathbf{x} = \vec{0}$ to compute $f(\gamma)$.

Let $V_\gamma(k)$ be the maximum contribution to the objective function of Problem (2.1) from all leaves in the subtree rooted at vertex k , given that feature k is demonstrated in-store. This requires $x_k = 1$ and restricts the objective function to summing over products in $L(k)$. Since $x_k = 1$, we know $x_{k'} = 1$ for all ancestors k' in $A(k)$. We use \mathcal{X}_k to denote the constraints of $\mathcal{X} \cap \{\mathbf{x} \mid x_{k'} = 1 \ \forall k' \in A(k)\}$. Since we only consider the contribution from the leaves in $L(k)$, we can focus on feasibility of the constraints in \mathcal{X} related to vertices in the subtree of k . By this definition, $V_\gamma(k)$ is:

$$V_\gamma(k) = \max_{\mathbf{x} \in \mathcal{X}_k} \sum_{i \in L(k)} (\pi_i - \gamma) w_i \cdot \prod_{k' \in A(i)} \delta_{k'}^{x_{k'}}.$$

If k is a leaf, then feasibility to \mathcal{X}_k requires $V_\gamma(k) = (\pi_k - \gamma) w_k \cdot \prod_{k' \in A(k)} \delta_{k'}$. If k is the root, then $V_\gamma(\text{root})$ optimizes Problem (2.1) over all feasible solutions in $\mathcal{X}_{\text{root}} = \mathcal{X} \setminus \{\vec{0}\}$. The objective value of the remaining solution $\mathbf{x} = \vec{0}$ is $\sum_{i=1}^n (\pi_i - \gamma) w_i$;

hence our parametrized problem can be written as:

$$f(\gamma) = \max \left\{ V_\gamma(\text{root}), \sum_{i=1}^n (\pi_i - \gamma) w_i \right\}.$$

To efficiently compute the value of $V_\gamma(\text{root})$, we construct a dynamic program that solves the tree from the leaves up to the root, where each $V_\gamma(k)$ is related to the values at its children: $V_\gamma(\ell(k))$ and $V_\gamma(r(k))$. Since $x_k = 1$, we consider three cases: *i*) $x_{\ell(k)} = x_{r(k)} = 1$, *ii*) $x_{\ell(k)} = 1, x_{r(k)} = 0$, or *iii*) $x_{\ell(k)} = 0, x_{r(k)} = 1$. In each of these cases, the objective function and constraints can be broken up by the subtrees of $\ell(k)$ and $r(k)$, and we can optimize each subtree separately.

In cases (*ii*) and (*iii*), we need to consider the contribution to the objective function from the leaves in the subtree of the undemonstrated child of k . Consider case (*ii*) where $x_{r(k)} = 0$. Feature $r(k)$ and its descendants are not demonstrated in-store, but feature k and all its ancestors are demonstrated. The preference weight of product i in $L(r(k))$ is:

$$w_i \cdot \prod_{k' \in A(i)} \delta_{k'}^{x_{k'}} = w_i \cdot \prod_{k' \in A(k)} \delta_{k'}.$$

The same analysis holds for $i \in L(\ell(k))$ if we consider case (*iii*). To simplify notation, let Δ_k denote the product of all $\delta_{k'}$ such that k' is an ancestor of k . This is defined recursively, with the base case $\Delta_{p(\text{root})} = 1$:

$$\Delta_k = \prod_{k' \in A(k)} \delta_{k'} = \Delta_{p(k)} \cdot \delta_k \quad \forall k = 1, \dots, 2n - 1.$$

For computational purpose, the size of Δ_k is still polynomial in the input sizes, as $\log \Delta_k = \sum_{k' \in A(k)} \log \delta_{k'} \leq n \cdot \max_{k'} \log \delta_{k'}$. Based on the three cases above, we can

rewrite $V_\gamma(k)$ as:

$$V_\gamma(k) = (\pi_k - \gamma)w_k\Delta_k \quad \forall k \in N,$$

$$V_\gamma(k) = \max \left\{ \begin{array}{l} V_\gamma(\ell(k)) + V_\gamma(r(k)), \\ V_\gamma(\ell(k)) + \Delta_k \cdot \sum_{i \in L(r(k))} (\pi_i - \gamma)w_i, \\ \Delta_k \cdot \sum_{i \in L(\ell(k))} (\pi_i - \gamma)w_i + V_\gamma(r(k)) \end{array} \right\} \quad \forall k \notin N.$$

The base cases are $k \in N$ and we solve the dynamic program from the leaves up to the root. For any fixed γ , computing $V_\gamma(\text{root})$ requires us to solve a dynamic program with $O(n)$ states and 3 decisions per state. We compare the value of $V_\gamma(\text{root})$ to the objective value when $\mathbf{x} = \vec{0}$, to obtain $f(\gamma) = \max\{V_\gamma(\text{root}), \sum_{i=1}^n (\pi_i - \gamma)w_i\}$. Hence, we can compute $f(\gamma)$ in $O(n)$ operations.

Finally, we consider the number of operations necessary to find γ^* such that $f(\gamma^*) = w_0\gamma^*$. The parametrized problem $f(\gamma)$ is monotone decreasing in γ , and one way to find our solution γ^* is via bisection search between an upper and lower bound on the online expected revenue. In order to bound the runtime of bisection search, we need to bound the smallest gap in expected revenue between two assortments, which is not a simple task. Another method is to apply Newton's method (Radzik (1998)). If the numerator and denominator of the online expected revenue can be written as linear functions, then the fixed point can be found in $O(n^2 \log^2 n)$ iterations of Newton's method because $\mathcal{X} \subseteq \{0, 1\}^{2n-1}$. The next lemma shows that the numerator and denominator of the online expected revenue can be written as linear functions of $\mathbf{x} \in \mathcal{X}$.

Lemma 1. *The preference weight of product i for an online customer, if she observes*

assortment \mathbf{x} , can be written as a linear function of $\mathbf{x} \in \mathcal{X}$:

$$v_i(\mathbf{x}) = w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k} = w_i \cdot \left(1 + \sum_{k \in A(i)} (\Delta_k - \Delta_{p(k)}) \cdot x_k \right).$$

Hence, we can write the online expected profit as:

$$\Pi^{ON}(\mathbf{x}) = \frac{\sum_{i=1}^n \pi_i w_i + \sum_{k=1}^{2n-1} \left(\sum_{i \in L(k)} \pi_i w_i \right) \cdot (\Delta_k - \Delta_{p(k)}) \cdot x_k}{w_0 + \sum_{i=1}^n w_i + \sum_{k=1}^{2n-1} \left(\sum_{i \in L(k)} w_i \right) \cdot (\Delta_k - \Delta_{p(k)}) \cdot x_k}.$$

The proof is deferred to Appendix A.1.

Theorem 1. *In the showroom setting of the OCA problem under the features tree model, where all customers use the physical store as a showroom to observe features and purchase ultimately from the online assortment, we can compute an optimal display assortment in $O(n^3 \log^2 n)$ operations.*

Proof. As stated previously, finding the optimal assortment in the showroom setting requires computing $f(\gamma)$ and finding γ^* such that $f(\gamma^*) = w_0 \gamma^*$. It takes $O(n)$ operations to compute $f(\gamma)$ for each possible value of γ . Using Newton's method, we can find γ^* in $O(n^2 \log^2 n)$ iterations of computing $f(\gamma)$. Hence the total number of operations is $O(n^3 \log^2 n)$. \square

2.4 General Setting

Individually, optimizing the online expected revenue (showroom setting) and the offline expected revenue can be done in polynomial time. Since the objective function of the general setting is a sum of fractions, one should suspect that the OCA problem under the features tree model is NP-hard when $q \in (0, 1)$, as is the case with many assortment optimization problems where the objective is a sum

of fractions. We prove this in the next proposition. The proof is in Appendix A.1.

Proposition 1. *The general setting of the OCA problem under the features tree model, where $q \in (0, 1)$, is NP-hard.*

In Subsection 2.4.1, we describe our FPTAS to compute an assortment guaranteeing $(1 - \epsilon)$ of the optimal expected revenue. In order to run our FPTAS, our algorithm requires us to solve a parametrized problem, which will be done via dynamic programming in Subsection 2.4.2.

2.4.1 FPTAS for General Setting

Instead of optimizing over the sum of two fractions, we will extend the mathematical program in the showroom setting to incorporate bounds on the numerator and denominator of the offline expected revenue. Suppose \mathbf{x}^* is the optimal solution to our problem and let $R^* = \sum_{i=1}^n \pi_i \hat{v}_i x_i^*$, $U^* = \hat{v}_0 + \sum_{i=1}^n \hat{v}_i x_i^*$. Denote the optimal expected revenue as $\Pi^* = \Pi(\mathbf{x}^*)$.

For a pair $(R, U) \geq (0, \hat{v}_0)$, consider the following problem which optimizes the online expected revenue over $\mathbf{x} \in \mathcal{X}$, subject to constraints on the numerator and denominator of the offline expected revenue based on inputs (R, U) :

$$g(R, U) = \max_{\mathbf{x} \in \mathcal{X}} \left\{ \frac{\sum_{i=1}^n \pi_i w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}}{w_0 + \sum_{i=1}^n w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}} \mid \sum_{i=1}^n \pi_i \hat{v}_i x_i \geq R, \hat{v}_0 + \sum_{i=1}^n \hat{v}_i x_i \leq U \right\}. \quad (2.2)$$

Solving Problem (2.2) at (R^*, U^*) gives us the characteristic vector of the optimal assortment. This is because \mathbf{x}^* is a feasible solution with objective value $\Pi^{ON}(\mathbf{x}^*)$. Any optimal solution \mathbf{x}' to Problem (2.2) at (R^*, U^*) would earn offline expected

revenue of $\Pi^{PHY}(\mathbf{x}') \geq R^*/U^*$ because \mathbf{x}' satisfies the two constraints above, and earn online expected revenue of $\Pi^{ON}(\mathbf{x}') \geq \Pi^{ON}(\mathbf{x}^*)$ because \mathbf{x}' is optimal.

The problem is that we do not know R^* or U^* , and it is not clear that we can solve Problem (2.2) efficiently. Our strategy is to apply a geometric grid to the possible values of R and U and approximately compute $g(R, U)$ within this grid. The optimal solution on this grid will give us a FPTAS to our problem.

To set up our grid, let $\underline{R} = \min_{i \in N} \pi_i \hat{v}_i$ and $\bar{R} = \max_{i \in N} \pi_i \hat{v}_i$. Similarly, let $\underline{U} = \min_{i \in N \cup \{0\}} \hat{v}_i$ and $\bar{U} = \max_{i \in N \cup \{0\}} \hat{v}_i$. Then \underline{R} and $n\bar{R}$ are the lower- and upper-bounds on the numerator of $\Pi^{PHY}(\cdot)$, assuming that at least one product is offered in-store. Similarly, \underline{U} and $(n+1)\bar{U}$ are the lower- and upper-bounds of the denominator of $\Pi^{PHY}(\cdot)$. Given $\epsilon > 0$, our grid \mathcal{K}_ϵ is constructed as:

$$\begin{aligned}\mathcal{K}_\epsilon^R &= \{\underline{R} \cdot (1 + \epsilon)^d \mid \underline{R} \cdot (1 + \epsilon)^d \leq (1 + \epsilon) \cdot n\bar{R}, d \in \mathbb{Z}_+\}, \\ \mathcal{K}_\epsilon^U &= \{\underline{U} \cdot (1 + \epsilon)^d \mid \underline{U} \cdot (1 + \epsilon)^d \leq (1 + \epsilon) \cdot (n+1)\bar{U}, d \in \mathbb{Z}_+\}, \text{ and} \\ \mathcal{K}_\epsilon &= \{(0, \hat{v}_0)\} \cup (\mathcal{K}_\epsilon^R \times \mathcal{K}_\epsilon^U).\end{aligned}$$

Our grid \mathcal{K}_ϵ has $O\left(\frac{\log n\bar{R}/\underline{R}}{\epsilon} \cdot \frac{\log(n+1)\bar{U}/\underline{U}}{\epsilon}\right)$ points. Let $\mathcal{K}_{\text{feas}} \subseteq \mathcal{K}_\epsilon$ denote the set of points $(R, U) \in \mathcal{K}_\epsilon$ such that Problem (2.2) is feasible and $g(R, U)$ is well-defined.

Proposition 2. *Consider an instance of the OCA problem under the features tree model with optimal expected revenue of Π^* and $\epsilon < 1/6$. For any $(R, U) \in \mathcal{K}_{\text{feas}}$, suppose we can compute a solution $\mathbf{x}^{R,U}$ achieving online expected revenue $\Pi^{ON}(\mathbf{x}^{R,U}) \geq g(R, U)$ and satisfying:*

$$\begin{aligned}\sum_{i=1}^n \pi_i \hat{v}_i x_i^{R,U} &\geq (1 - 2\epsilon)R \\ \hat{v}_0 + \sum_{i=1}^n \hat{v}_i x_i^{R,U} &\leq (1 + 2\epsilon)U \\ \mathbf{x}^{R,U} &\in \mathcal{X}.\end{aligned}$$

Let $\bar{\mathbf{x}} = \arg \max_{(R,U) \in \mathcal{K}_{\text{feas}}} \Pi(\mathbf{x}^{R,U})$. Then $\Pi(\bar{\mathbf{x}}) \geq (1 - 6\epsilon)\Pi^*$.

Proof. If Problem (2.2) is feasible at (R, U) , then $\mathbf{x}^{R,U}$ generates an expected revenue of

$$\begin{aligned} \Pi(\mathbf{x}^{R,U}) &\geq q \cdot \Pi^{ON}(\mathbf{x}^{R,U}) + (1 - q) \cdot \frac{(1 - 2\epsilon)R}{(1 + 2\epsilon)U} \\ &\geq \frac{1 - 2\epsilon}{1 + 2\epsilon} \cdot \left(q \cdot \Pi^{ON}(\mathbf{x}^{R,U}) + (1 - q) \cdot \frac{R}{U} \right). \end{aligned}$$

If the optimal assortment is $\mathbf{x}^* = \vec{0}$, then our solution to Problem (2.2) at $(0, \hat{v}_0) \in \mathcal{K}_\epsilon$ is optimal because $\mathbf{x}^{0,\hat{v}_0} = \vec{0}$ is the unique solution at this grid point. Otherwise, there exists $(R', U') \in \mathcal{K}_\epsilon$ such that $R' \leq R^* < (1 + \epsilon)R'$ and $U' \leq U^* < (1 + \epsilon)U'$. We can use these points to upper-bound our optimal expected revenue. Since Problem (2.2) at $(R', (1 + \epsilon)U')$ is a relaxation of Problem (2.2) at (R^*, U^*) , Problem (2.2) must be feasible at $(R', (1 + \epsilon)U')$ and hence $(R', (1 + \epsilon)U') \in \mathcal{K}_{\text{feas}}$.

$$\begin{aligned} \Pi^* &= q \cdot g(R^*, U^*) + (1 - q) \cdot \frac{R^*}{U^*} \\ &\leq q \cdot g(R', (1 + \epsilon)U') + (1 - q) \cdot \frac{(1 + \epsilon)R'}{U'} \\ &\leq q \cdot \Pi^{ON}(\mathbf{x}^{R', (1 + \epsilon)U'}) + (1 - q) \cdot (1 + \epsilon)^2 \cdot \frac{R'}{(1 + \epsilon)U'} \\ &\leq (1 + \epsilon)^2 \left(q \cdot \Pi^{ON}(\mathbf{x}^{R', (1 + \epsilon)U'}) + (1 - q) \cdot \frac{R'}{(1 + \epsilon)U'} \right) \\ &\leq \frac{(1 + \epsilon)^2(1 + 2\epsilon)}{1 - 2\epsilon} \Pi(\mathbf{x}^{R', (1 + \epsilon)U'}) \\ &\leq \frac{1}{1 - 6\epsilon} \Pi(\bar{\mathbf{x}}). \end{aligned}$$

The first inequality holds because we relaxed the right side of Problem (2.2). The second inequality holds by assumption, and the fourth inequality holds by the inequalities at the beginning of the proof. The last inequality holds by simplifying the ratio of ϵ 's and the choice of $\bar{\mathbf{x}}$. \square

Our main challenge is to efficiently compute $\mathbf{x}^{R,U}$ satisfying the assumptions of Proposition 2 for each $(R, U) \in \mathcal{K}_{\text{feas}}$. If we do the standard parametrization of the objective function, then we are looking at Problem (2.1) from the showroom setting, subject to the two additional knapsack-like constraints of Problem (2.2). This structure suggests rounding inputs $\pi_i \hat{v}_i$ and \hat{v}_i based on R and U so that we can extend our dynamic program from the showroom setting to include the two knapsack-like constraints. For $i \in N$, we round the parameters as follows:

$$\tilde{\pi}_i^R = \left\lfloor \frac{\pi_i \hat{v}_i}{\epsilon R/n} \right\rfloor, \quad \text{and} \quad \tilde{v}_i^U = \left\lfloor \frac{\hat{v}_i}{\epsilon U/(n+1)} \right\rfloor.$$

We also round the offline preference weight of the no-purchase option and define Y_1, Y_2 as the polynomial-sized bounds on the rounded numerators and denominators of the offline expected revenue:

$$\tilde{v}_0^U = \left\lfloor \frac{\hat{v}_0}{\epsilon U/(n+1)} \right\rfloor, \quad Y_1 = \left\lfloor \frac{n}{\epsilon} \right\rfloor - n, \quad \text{and} \quad Y_2 = \left\lfloor \frac{n+1}{\epsilon} \right\rfloor + (n+1).$$

For each pair (R, U) , we parametrize Problem (2.2) and use the rounded constraints to obtain Problem (2.3) with value $\tilde{f}(\gamma, R, U)$.

$$\tilde{f}(\gamma, R, U) = \max_{\mathbf{x} \in \mathcal{X}} \left\{ \sum_{i=1}^n (\pi_i - \gamma) w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k} \left| \sum_{i=1}^n \tilde{\pi}_i^R x_i \geq Y_1, \quad \tilde{v}_0^U + \sum_{i=1}^n \tilde{v}_i^U x_i \leq Y_2 \right. \right\}. \quad (2.3)$$

Lemma 2. *Suppose \mathbf{x} is a feasible solution to Problem (2.2) at (R, U) . Then \mathbf{x} is a feasible solution to Problem (2.3) at (γ, R, U) for any γ . Furthermore, any feasible \mathbf{x} to Problem (2.3) satisfies $\sum_{i=1}^n \pi_i \hat{v}_i x_i \geq (1 - 2\epsilon)R$ and $\hat{v}_0 + \sum_{i=1}^n \hat{v}_i x_i \leq (1 + 2\epsilon)U$.*

Proof. If \mathbf{x} is feasible to Problem (2.2) at (R, U) , then $\mathbf{x} \in \mathcal{X}$, $\sum_{i=1}^n \pi_i \hat{v}_i x_i \geq R$, and $\hat{v}_0 + \sum_{i=1}^n \hat{v}_i x_i \leq U$. We check for feasibility of the two knapsack-like constraints in Problem (2.3). Using the modified values for the numerator, we obtain the

following chain of inequalities:

$$\sum_{i=1}^n \tilde{\pi}_i^R x_i = \sum_{i=1}^n \left\lfloor \frac{\pi_i \hat{v}_i}{\epsilon R/n} \right\rfloor x_i \geq \sum_{i=1}^n \left(\frac{\pi_i \hat{v}_i}{\epsilon R/n} - 1 \right) x_i \geq \frac{R}{\epsilon R/n} - \sum_{i=1}^n x_i \geq \left\lfloor \frac{n}{\epsilon} \right\rfloor - n = Y_1.$$

Similarly, we obtain the following chain of inequalities for the denominator:

$$\begin{aligned} \tilde{v}_0^U + \sum_{i=1}^n \tilde{v}_i^U x_i &= \left\lfloor \frac{\hat{v}_0}{\epsilon U/(n+1)} \right\rfloor + \sum_{i=1}^n \left\lfloor \frac{\hat{v}_i}{\epsilon U/(n+1)} \right\rfloor x_i \\ &\leq \frac{\hat{v}_0}{\epsilon U/(n+1)} + 1 + \sum_{i=1}^n \left(\frac{\hat{v}_i}{\epsilon U/(n+1)} + 1 \right) x_i \\ &\leq \frac{U}{\epsilon U/(n+1)} + 1 + \sum_{i=1}^n x_i \leq \left\lfloor \frac{n+1}{\epsilon} \right\rfloor + 1 + n = Y_2. \end{aligned}$$

Hence \mathbf{x} is feasible to Problem (2.3) at (γ, R, U) .

Next, if \mathbf{x} is feasible to Problem (2.3) at (γ, R, U) , then we have the following chain of inequalities:

$$\sum_{i=1}^n \pi_i v_i x_i \geq \frac{\epsilon R}{n} \cdot \sum_{i=1}^n \tilde{\pi}_i^R x_i \geq \frac{\epsilon R}{n} \cdot \left(\left\lfloor \frac{n}{\epsilon} \right\rfloor - n \right) \geq R - \frac{\epsilon R}{n} - \epsilon R \geq (1 - 2\epsilon)R.$$

Similarly, we have the following chain of inequalities when considering the denominator:

$$\begin{aligned} \hat{v}_0 + \sum_{i=1}^n \hat{v}_i x_i &\leq \frac{\epsilon U}{n+1} \left(\tilde{v}_0^U + \sum_{i=1}^n \tilde{v}_i^U x_i \right) \\ &\leq \frac{\epsilon U}{n+1} \cdot \left(\left\lfloor \frac{n+1}{\epsilon} \right\rfloor + 1 + n \right) \\ &\leq U + \frac{\epsilon U}{n+1} + \epsilon U \leq (1 + 2\epsilon)U. \end{aligned}$$

□

Corollary 1. Suppose $(R, U) \in \mathcal{K}_{\text{feas}}$. Then there exists $\gamma^{R,U}$ such that $\tilde{f}(\gamma^{R,U}, R, U) = w_0 \gamma^{R,U}$. Furthermore, the optimal solution $\mathbf{x}^{R,U}$ of Problem (2.3) with inputs $(\gamma^{R,U}, R, U)$ satisfies $\sum_{i=1}^n \pi_i \hat{v}_i x_i^{R,U} \geq (1 - 2\epsilon)R$ and $\hat{v}_0 + \sum_{i=1}^n \hat{v}_i x_i^{R,U} \leq (1 + 2\epsilon)U$, and ensures that $\Pi^{ON}(\mathbf{x}^{R,U}) \geq g(R, U)$.

Proof. By Lemma 2, Problem (2.3) has a feasible solution because Problem (2.2) is feasible at (R, U) . The value of $\tilde{f}(\gamma, R, U)$ is decreasing in γ and $\tilde{f}(0, R, U) > 0$, so there exists $\gamma^{R,U}$ such that $\tilde{f}(\gamma^{R,U}, R, U) = w_0 \gamma^{R,U}$, with corresponding optimal solution $\mathbf{x}^{R,U}$ to Problem (2.3) at $(\gamma^{R,U}, R, U)$. Furthermore, we can rearrange the objective function to observe that $\Pi^{ON}(\mathbf{x}^{R,U}) = \gamma^{R,U}$. In particular, by definition of $\gamma^{R,U}$, we have:

$$\sum_{i=1}^n (\pi_i - \gamma^{R,U}) w_i \cdot \prod_{k \in A(i)} \delta_k^{\gamma^{R,U}} = w_0 \gamma^{R,U}.$$

By rearranging, we obtain:

$$\gamma^{R,U} = \frac{\sum_{i=1}^n \pi_i w_i \cdot \prod_{k \in A(i)} \delta_k^{\gamma^{R,U}}}{w_0 + \sum_{i=1}^n w_i \cdot \prod_{k \in A(i)} \delta_k^{\gamma^{R,U}}} = \Pi^{ON}(\mathbf{x}^{R,U}).$$

Next, suppose \mathbf{x}^g optimizes Problem (2.2) with inputs (R, U) . Applying Lemma 2 again tells us that \mathbf{x}^g is feasible to Problem (2.3) with inputs $(\gamma^{R,U}, R, U)$. Hence, its objective value in Problem (2.3) is upper-bounded by $w_0 \gamma^{R,U}$

$$\sum_{i=1}^n (\pi_i - \gamma^{R,U}) w_i \cdot \prod_{k \in A(i)} \delta_k^{\mathbf{x}^g} \leq w_0 \gamma^{R,U}.$$

By rearranging, we observe:

$$\gamma^{R,U} \geq \frac{\sum_{i=1}^n \pi_i w_i \cdot \prod_{k \in A(i)} \delta_k^{\mathbf{x}^g}}{w_0 + \sum_{i=1}^n w_i \cdot \prod_{k \in A(i)} \delta_k^{\mathbf{x}^g}} = g(R, U).$$

Therefore, $g(R, U) \leq \Pi^{ON}(\mathbf{x}^{R,U})$. □

In summary, we set up our FPTAS for the general setting by creating a geometric grid on the numerator and denominator of the offline expected revenue. For each point (R, U) on the grid, a feasible solution to Problem (2.2) has offline expected revenue greater or equal to R/U . However, Problem (2.2) is hard to solve because of the knapsack-like constraints. Instead, our FPTAS only requires

a slightly perturbed solution to Problem (2.2) which satisfies the conditions of Proposition 2. Then Lemma 2 and Corollary 1 tells us that we can parametrize a rounded version of Problem (2.2) to arrive at Problem (2.3); the optimal solution $\mathbf{x}^{R,U}$ to the fixed point $\gamma^{R,U}$ of Problem (2.3) satisfies the conditions of Proposition 2. Since the parameters in the constraints of Problem (2.3) are integers of size $O(n/\epsilon)$, we can find its optimal solution via dynamic programming, which is described in the next subsection.

2.4.2 Solving the Parametrized Problem at a Grid-point

In order to solve Problem (2.3) for inputs (γ, R, U) , we set up our dynamic program following the structure of Section 2.3. Before proceeding with the value function, there are two cases of $(R, U) \in \mathcal{K}_\epsilon$ that we have to consider. If $(R, U) = (0, \hat{v}_0)$, then the only feasible solution to Problem (2.3) is $\mathbf{x} = \vec{0}$ because $\tilde{v}_i^U \geq 1$ for all $i \in N$. Hence, the characteristic vector of any non-empty assortment violates the second constraint of Problem (2.3) and the value of $g(0, \hat{v}_0)$ can be computed directly.

On the other hand, if $(R, U) \neq (0, \hat{v}_0)$, then $R > 0$ and $\mathbf{x} = \vec{0}$ is infeasible to the first constraint of Problem (2.3). The feasible region of Problem (2.3) is contained in $\mathcal{X} \setminus \{\vec{0}\}$, which is equal to $\mathcal{X}_{\text{root}}$. We proceed to solve Problem (2.3) for $(R, U) \in \mathcal{K}_\epsilon \setminus \{(0, \hat{v}_0)\}$ via dynamic programming.

Like the showroom setting, we set up a value function which considers the maximum contribution from the leaves in the subtree rooted at vertex k , given that feature k is displayed. In addition, we need to include two more states in our value function to recognize the impact of $L(k)$ to the two knapsack-like

constraints. Let $\tilde{V}_\gamma^{R,U}(k, y_1, y_2)$ be the maximum contribution from leaves in $L(k)$ when feature k is demonstrated in-store, subject to the constraints of \mathcal{X}_k and the leaves in $L(k)$ contributing at least y_1 to the first constraint of Problem (2.3) and at most y_2 to the second constraint. Mathematically, this is represented by:

$$\tilde{V}_\gamma^{R,U}(k, y_1, y_2) = \max_{\mathbf{x} \in \mathcal{X}_k} \left\{ \sum_{i \in L(k)} (\pi_i - \gamma) w_i \cdot \prod_{k' \in A(i)} \delta_{k'}^{x_{k'}} \mid \sum_{i \in L(k)} \tilde{\pi}_i^R x_i \geq y_1, \sum_{i \in L(k)} \tilde{v}_i^U x_i \leq y_2 \right\}.$$

By definition of our value function, the value of our parametrized problem at $(R, U) \in \mathcal{K}_\epsilon \setminus \{(0, \hat{v}_0)\}$ is:

$$\tilde{f}(\gamma, R, U) = \tilde{V}_\gamma^{R,U}(\text{root}, Y_1, Y_2 - \tilde{v}_0^U).$$

There are two details that may not be obvious at first glance. First, the third input to the value function at the root is $Y_2 - \tilde{v}_0^U$ and not Y_2 . Since the no-purchase option is not a decision, we exclude the preference weight of the no-purchase option from the second constraint of $\tilde{V}_\gamma^{R,U}(k, y_1, y_2)$ at a subtree. Instead, \tilde{v}_0^U is a constant that can be moved to the right side of the second constraint in Problem (2.3). Second, when we compute $f(\gamma)$ in Problem (2.1) of Section 2.3, we compare the result of the dynamic program at $V_\gamma(\text{root})$ to the objective function at $\mathbf{x} = \vec{0}$ because the feasible region $\mathcal{X}_{\text{root}}$ excludes the empty assortment. As stated previously, we do not have to consider $\mathbf{x} = \vec{0}$ in Problem (2.3) because the feasible region is reduced to $\mathcal{X}_{\text{root}}$ when $(R, U) \neq (0, \hat{v}_0)$.

We can rewrite $\tilde{V}_\gamma^{R,U}(k, y_1, y_2)$ based on the contributions from the children of k . Since $x_k = 1$, there are three cases that we consider: *i*) $x_{\ell(k)} = x_{r(k)} = 1$, *ii*) $x_{\ell(k)} = 1, x_{r(k)} = 0$, and *iii*) $x_{\ell(k)} = 0, x_{r(k)} = 1$. In the first case, we look at all possible splits of the required y_1, y_2 across the trees rooted at $\ell(k)$ and $r(k)$. In the second case, since $x_{r(k)} = 0$, the leaves in $L(r(k))$ cannot contribute to the two constraints and the required y_1, y_2 have to be satisfied on the left subtree. A

similar argument applies to the third case. Hence, $\tilde{V}_\gamma^{R,U}(k, y_1, y_2)$ can be rewritten as the following dynamic program where $y_1 \in \{0, \dots, Y_1\}$ and $y_2 \in \{0, \dots, Y_2 - \tilde{v}_0^U\}$:

$$\tilde{V}_\gamma^{R,U}(k, y_1, y_2) = \begin{cases} (\pi_k - \gamma)w_k\Delta_k & \text{if } y_1 \leq \tilde{\pi}_k^R, y_2 \geq \tilde{v}_k^U \\ -\infty & \text{otherwise} \end{cases} \quad \forall k \in N,$$

$$\tilde{V}_\gamma^{R,U}(k, y_1, y_2) = \max \left\{ \begin{array}{l} \max_{\substack{0 \leq x_1 \leq y_1 \\ 0 \leq x_2 \leq y_2}} \tilde{V}_\gamma^{R,U}(\ell(k), x_1, x_2) + \tilde{V}_\gamma^{R,U}(r(k), y_1 - x_1, y_2 - x_2), \\ \tilde{V}_\gamma^{R,U}(\ell(k), y_1, y_2) + \Delta_k \cdot \sum_{i \in L(r(k))} (\pi_i - \gamma)w_i, \\ \Delta_k \cdot \sum_{i \in L(\ell(k))} (\pi_i - \gamma)w_i + \tilde{V}_\gamma^{R,U}(r(k), y_1, y_2) \end{array} \right\} \quad \forall k \notin N.$$

To summarize this section, our FPTAS for finding assortment $\bar{\mathbf{x}}$ such that $\Pi(\bar{\mathbf{x}}) \geq (1 - \epsilon)\Pi^*$ is presented in Algorithm 1. The set $\tilde{\mathcal{K}}_{\text{feas}}$ on the second line contains all (R, U) such that Problem (2.3) is feasible at $(0, R, U)$. By Lemma 2, feasibility of Problem (2.2) implies feasibility of Problem (2.3), so we have $\mathcal{K}_{\text{feas}} \subseteq \tilde{\mathcal{K}}_{\text{feas}}$. The while loop implements Newton's method to find γ such that $\tilde{f}(\gamma, R, U) = w_0\gamma$. Our returned solution $\bar{\mathbf{x}}$ satisfies $\Pi(\bar{\mathbf{x}}) \geq \max_{(R,U) \in \mathcal{K}_{\text{feas}}} \Pi(\mathbf{x}^{R,U}) \geq (1 - \epsilon)\Pi^*$ where the first inequality is due to $\mathcal{K}_{\text{feas}} \subseteq \tilde{\mathcal{K}}_{\text{feas}}$ and the second inequality is due to Proposition 2.

Theorem 2. *Suppose Π^* is the optimal expected revenue of the OCA problem under the features tree model. For any $\epsilon \in (0, 1)$, there exists an algorithm that finds an in-store assortment \mathbf{x} such that $\Pi(\mathbf{x}) \geq (1 - \epsilon)\Pi^*$ in $O\left(\frac{n^7 \log^2 n \log n \bar{R}/R \cdot \log(n+1)\bar{U}/U}{\epsilon^6}\right)$ operations.*

Proof. Corollary 1 gives an $\mathbf{x}^{R,U}$ that satisfies Proposition 2 whenever Problem (2.2) is feasible at (R, U) . By Proposition 2, the solution $\bar{\mathbf{x}} = \arg \max_{(R,U) \in \mathcal{K}_{\text{feas}}} \Pi(\mathbf{x}^{R,U})$ satisfies $\Pi(\bar{\mathbf{x}}) \geq (1 - 6\epsilon')\Pi^*$ for $\epsilon' < 1/6$. So given any $\epsilon \in (0, 1)$, set $\epsilon' = \epsilon/6$ to get a $(1 - \epsilon)$ -approximation.

Algorithm 1: FPTAS to find an assortment $\bar{\mathbf{x}}$ such that $\Pi(\bar{\mathbf{x}}) \geq (1 - \epsilon)\Pi^*$

Input : Instance of OCA under the features tree model with desired accuracy ϵ

Output: Characteristic vector $\bar{\mathbf{x}}$ such that $\Pi(\bar{\mathbf{x}}) \geq (1 - \epsilon)\Pi^*$

Set $\epsilon \leftarrow \epsilon/6$ and construct \mathcal{K}_ϵ ;

Initialize grid $\tilde{\mathcal{K}}_{\text{feas}} \leftarrow \{(0, \hat{v}_0)\}$;

Set $\mathbf{x}^{0, \hat{v}_0} \leftarrow \vec{0}$;

for $(R, U) \in \mathcal{K}_\epsilon \setminus \{(0, \hat{v}_0)\}$ **do**

 Set $\gamma \leftarrow 0$;

 Solve Problem (2.3) for optimal \mathbf{x} , if feasible ;

if Problem (2.3) is feasible **then**

 Update $\tilde{\mathcal{K}}_{\text{feas}} \leftarrow \tilde{\mathcal{K}}_{\text{feas}} \cup \{(R, U)\}$;

while $\tilde{f}(\gamma, R, U) > w_0 \gamma$ **do**

 Update $\gamma \leftarrow \Pi^{ON}(\mathbf{x})$;

 Resolve Problem (2.3) for optimal \mathbf{x} and value $\tilde{f}(\gamma, R, U)$;

end

 Set $\mathbf{x}^{R, U} \leftarrow \mathbf{x}$;

end

Return $\bar{\mathbf{x}} = \arg \max_{(R, U) \in \tilde{\mathcal{K}}_{\text{feas}}} \Pi(\mathbf{x}^{R, U})$.

There are $O\left(\frac{\log n \bar{R}/R}{\epsilon} \cdot \frac{\log(n+1) \bar{U}/U}{\epsilon}\right)$ pairs of $(R, U) \in \mathcal{K}_\epsilon$ for which we have to compute $g(R, U)$ or determine that it is infeasible. If Problem (2.3) is feasible at $(0, R, U)$, then we run Newton's method to find $\gamma^{R, U}$ such that $\tilde{f}(\gamma^{R, U}, R, U) = w_0 \gamma^{R, U}$, which takes $O(n^2 \log^2 n)$ iterations. In total, the number of times that we need to compute $\tilde{V}_\gamma^{R, U}(\text{root}, Y_1, Y_2 - \tilde{v}_0^U)$ is $O\left(\frac{n^2 \log^2 n \log n \bar{R}/R \cdot \log(n+1) \bar{U}/U}{\epsilon^2}\right)$.

Finally, we need to compute $\tilde{V}_\gamma^{R, U}(\text{root}, Y_1, Y_2 - \tilde{v}_0^U)$ in order to solve Problem (2.3) at (γ, R, U) . We compute $\tilde{V}_\gamma^{R, U}(k, y_1, y_2)$ for $O(n)$ vertices in T and $O(n^2/\epsilon^2)$ pairs of (y_1, y_2) per vertex. For each state, we consider at most $O(n^2/\epsilon^2)$ ways to split (y_1, y_2) over the left and right children of k . Hence there are $O(n^5/\epsilon^4)$ operations to obtain $\tilde{V}_\gamma^{R, U}(\text{root}, Y_1, Y_2 - \tilde{v}_0^U)$. Therefore, we have a total runtime of $O\left(\frac{n^7 \log^2 n \log n \bar{R}/R \cdot \log(n+1) \bar{U}/U}{\epsilon^6}\right)$. \square

In Appendix A.2, we reduce the runtime by a factor of $O(n^2/\epsilon^2)$ by solving Problem (2.3) via a fancier constrained longest path construction in a directed acyclic graph. In Theorem 9 in the appendix, we compute the FPTAS assortment in $O\left(\frac{n^5 \log^2 n \log n \bar{R}/\underline{R} \cdot \log(n+1) \bar{U}/\underline{U}}{\epsilon^4}\right)$ operations.

2.5 Upper Bound on Optimal Expected Revenue

In order to evaluate the performance of our FPTAS in practice, we need a reasonable upper-bound on the optimal expected revenue without enumerating all possible assortments. We use the grid \mathcal{K}_ϵ from Section 2.4 to consider a simpler problem, but we can construct it with a smaller $\epsilon > 0$ in order to tighten our upper-bound.

Let $g^{\text{LP}}(R, U)$ be the value of Problem (2.2) with the integrality constraint relaxed. Let \mathcal{X}^{LP} denote the linear realization of \mathcal{X} where we replace $\mathbf{x} \in \{0, 1\}^{2n-1}$ with $0 \leq \mathbf{x} \leq 1$. Then $g^{\text{LP}}(R, U)$ is:

$$g^{\text{LP}}(R, U) = \max_{\mathbf{x} \in \mathcal{X}^{\text{LP}}} \left\{ \frac{\sum_{i=1}^n \pi_i w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}}{w_0 + \sum_{i=1}^n w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}} \mid \sum_{i=1}^n \pi_i \hat{v}_i x_i \geq R, \hat{v}_0 + \sum_{i=1}^n \hat{v}_i x_i \leq U \right\}. \quad (2.4)$$

We explain how $g^{\text{LP}}(R, U)$ can be computed for any $R \geq 0$, $U \geq \hat{v}_0$ after the following theorem, which describes the upper-bound on the optimal expected revenue Π^* . Recall that if we have the optimal characteristic vector \mathbf{x}^* , then $R^* = \sum_{i=1}^n \pi_i \hat{v}_i x_i^*$ and $U^* = \hat{v}_0 + \sum_{i=1}^n \hat{v}_i x_i^*$.

Proposition 3. *Define an upper-bound at grid point $(R, U) \in \mathcal{K}_{\text{feas}}$ as $\Pi^{\text{LP}}(R, U) = q \cdot g^{\text{LP}}(R, U) + (1 - q) \cdot \frac{(1+\epsilon)^2 R}{U}$. Then $\max_{(R, U) \in \mathcal{K}_{\text{feas}}} \Pi^{\text{LP}}(R, U) \geq \Pi^*$.*

Proof. There exists $(R', U') \in \mathcal{K}_\epsilon$ such that $R' \leq R^* < (1 + \epsilon)R'$ and $U' \leq U^* < (1 + \epsilon)U'$. From the proof of Proposition 2, we know $(R', (1 + \epsilon)U') \in \mathcal{K}_{\text{feas}}$. Since Problem (2.4) is a relaxation of Problem (2.2), we have:

$$g^{\text{LP}}(R', (1 + \epsilon)U') \geq g(R', (1 + \epsilon)U') \geq g(R^*, U^*).$$

Furthermore, $(1 + \epsilon)R'/U' \geq R^*/U^*$, and we obtain the following bound:

$$\begin{aligned} \max_{(R, U) \in \mathcal{K}_{\text{feas}}} \Pi^{\text{LP}}(R, U) &\geq \Pi^{\text{LP}}(R', (1 + \epsilon)U') \\ &= q \cdot g^{\text{LP}}(R', (1 + \epsilon)U') + (1 - q) \cdot \frac{(1 + \epsilon)^2 R'}{(1 + \epsilon)U'} \\ &\geq q \cdot g(R^*, U^*) + (1 - q) \cdot \frac{R^*}{U^*} = \Pi^*. \end{aligned}$$

□

To compute $g^{\text{LP}}(R, U)$, we consider the parametrized form of Problem (2.4) with optimal objective value $f^{\text{LP}}(\gamma, R, U)$. We can also apply Lemma 1 so that the objective function of the parametrized problem is linear in \mathbf{x} . The parametrized problem with optimal objective value $f^{\text{LP}}(\gamma, R, U)$ is:

$$\begin{aligned} f^{\text{LP}}(\gamma, R, U) = \max_{\mathbf{x} \in \mathcal{X}^{\text{LP}}} &\left\{ \sum_{i=1}^n (\pi_i - \gamma) w_i + \sum_{k=1}^{2n-1} \left(\sum_{i \in L(k)} (\pi_i - \gamma) w_i \right) \cdot (\Delta_k - \Delta_{p(k)}) \cdot x_k \right. \\ &\left. \sum_{i=1}^n \pi_i \hat{v}_i x_i \geq R, \hat{v}_0 + \sum_{i=1}^n \hat{v}_i x_i \leq U \right\}. \end{aligned} \quad (2.5)$$

Claim 1 applies with a slight modification, hence $\gamma = g^{\text{LP}}(R, U)$ if and only if $f^{\text{LP}}(\gamma, R, U) = w_0 \gamma$.

Problem (2.5) is a linear program in \mathbf{x} , so we can take the dual. The first term in the objective function, $\sum_{i=1}^n (\pi_i - \gamma) w_i$, is a constant and needs to be added back onto the dual's objective function. We use dual variables β_1, β_2 for the two

knapsack-like constraints of Problem (2.5). Dual variable y_k is used for the first and second constraint of \mathcal{X}_{LP} relating vertex k to its parent, and dual variable z_k is used for the third constraint relating vertex k to its children. Finally, we use dual variable u_k for the upper-bound on x_k . Then the dual linear program is:

$$\min \sum_{i=1}^n (\pi_i - \gamma) w_i + \sum_{k=1}^{2n-1} u_k - R\beta_1 + (U - \hat{v}_0)\beta_2$$

subject to

$$\begin{aligned} -\pi_k \hat{v}_k \beta_1 + \hat{v}_k \beta_2 + y_k - z_{p(k)} + u_k &\geq (\pi_k - \gamma) w_k \cdot (\Delta_k - \Delta_{p(k)}) & \forall k \in N, \\ y_k - y_{\ell(k)} - y_{r(k)} + z_k - z_{p(k)} + u_k &\geq \left(\sum_{i \in L(k)} (\pi_i - \gamma) w_i \right) \cdot (\Delta_k - \Delta_{p(k)}) & \forall k \notin N \cup \{\text{root}\}, \\ -y_{\ell(\text{root})} - y_{r(\text{root})} + z_{\text{root}} + u_{\text{root}} &\geq \left(\sum_{i \in N} (\pi_i - \gamma) w_i \right) \cdot (\Delta_k - 1) & , \\ \mathbf{y}, \mathbf{z}, \mathbf{u}, \boldsymbol{\beta} &\geq 0. \end{aligned}$$

Let \mathcal{Y} denote the set of constraints in the dual linear program. By strong duality, the dual linear program has optimal objective value $f^{\text{LP}}(\gamma, R, U)$ if Problem (2.5) is feasible. Since the dual is linear in γ , we can treat γ as a variable. We know $\gamma = g^{\text{LP}}(R, U)$ if and only if $f^{\text{LP}}(\gamma, R, U) = w_0 \gamma$, so we add an extra constraint to the dual to set the objective function equal to $w_0 \gamma$. Finally, we can scale the objective function of the dual by $1/w_0$ to solve directly for the desired γ . Our resulting problem is:

$$\min_{(\mathbf{y}, \mathbf{z}, \mathbf{u}, \boldsymbol{\beta}) \in \mathcal{Y}} \left\{ \gamma \left| \left(w_0 + \sum_{i=1}^n w_i \right) \gamma - \sum_{k=1}^{2n-1} u_k + R\beta_1 - (U - \hat{v}_0)\beta_2 = \sum_{i=1}^n \pi_i w_i, \gamma \geq 0 \right. \right\}. \quad (2.6)$$

Hence $g^{\text{LP}}(R, U)$ can be computed efficiently by solving Problem (2.6) for all $(R, U) \in \mathcal{K}_{\text{feas}}$. The constraint $\gamma \geq 0$ does not modify the feasible region when $(R, U) \in \mathcal{K}_{\text{feas}}$, because $g^{\text{LP}}(R, U) \geq g(R, U) > 0$. In particular, we know $(R, U) \notin \mathcal{K}_{\text{feas}}$ if Problem (2.6) has optimal objective value of 0 or is infeasible.

In the former case, the dual linear program of Problem (2.5) has non-positive optimal objective value, because the solution $(\mathbf{y}, \mathbf{z}, \mathbf{u}, \beta)$ of Problem (2.6) is feasible to the dual at $\gamma = 0$ with objective value 0. This contradicts $f^{\text{LP}}(0, R, U) > 0$ whenever $(R, U) \in \mathcal{K}_{\text{feas}}$.

To summarize this section, we present the upper-bound computation in Algorithm 2. Similar to Algorithm 1, we initialize a set \mathcal{K}_{LP} in the second line which contains all (R, U) such that Problem (2.6) is feasible and returns a positive value, so that $\mathcal{K}_{\text{feas}} \subseteq \mathcal{K}_{\text{LP}}$. The solution $\bar{\Pi}^{\text{LP}}$ returned by Algorithm 2 satisfies $\bar{\Pi}^{\text{LP}} \geq \max_{(R, U) \in \mathcal{K}_{\text{feas}}} \Pi^{\text{LP}}(R, U) \geq \Pi^*$ where the first inequality is due to $\mathcal{K}_{\text{feas}} \subseteq \mathcal{K}_{\text{LP}}$ and the second inequality is due to Proposition 3.

Algorithm 2: Upper-bound to measure the practical performance of our FPTAS

Input : Instance of OCA under the features tree model with desired accuracy ϵ

Output: Upper-bound value $\bar{\Pi}$ such that $\bar{\Pi} \geq \Pi^*$

Construct \mathcal{K}_ϵ ;

Initialize grid $\mathcal{K}_{\text{LP}} \leftarrow \{(0, \hat{v}_0)\}$;

Set $\Pi^{\text{LP}}(0, \hat{v}_0) \leftarrow q \cdot \frac{\sum_{i=1}^n \pi_i w_i}{w_0 + \sum_{i=1}^n w_i}$;

for $(R, U) \in \mathcal{K}_\epsilon \setminus \{(0, \hat{v}_0)\}$ **do**

 Solve Problem (2.6) at (R, U) for γ ;

if Problem (2.6) is feasible and $\gamma > 0$ **then**

 Update $\mathcal{K}_{\text{LP}} \leftarrow \mathcal{K}_{\text{LP}} \cup \{(R, U)\}$;

 Set $g^{\text{LP}}(R, U) = \gamma$ and compute $\Pi^{\text{LP}}(R, U)$;

end

Return $\bar{\Pi}^{\text{LP}} = \arg \max_{(R, U) \in \mathcal{K}_{\text{LP}}} \Pi^{\text{LP}}(R, U)$.

2.6 Extensions

In this section, we allow the retailer to choose his online assortment as well as his in-store assortment. We discuss the extensions with respect to the showroom setting. The techniques can be applied to the dynamic program in Subsection 2.4.2 if we consider the general setting instead. Let S^{ON} denote the online assortment and S^{PHY} denote the in-store assortment.

2.6.1 Related Online and In-store Assortments

In the first variation, the physical store shows a subset of the products in the online store. Many retailers encourage customers to order online if their style is not available in-store. We introduce binary variables y_i for $i \in N$, where $y_i = 1$ if product i is offered online and 0 otherwise. Since $S^{PHY} \subseteq S^{ON}$, we require $y_i \geq x_i$. The mathematical program for this setting is:

$$\max_{\substack{\mathbf{x} \in \mathcal{X} \\ \mathbf{y} \in \{0,1\}^n}} \left\{ \frac{\sum_{i=1}^n \pi_i w_i y_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}}{w_0 + \sum_{i=1}^n w_i y_i \cdot \prod_{k \in A(i)} \delta_k^{x_k}} \mid x_i \leq y_i \ \forall i \in N \right\}. \quad (2.7)$$

We can parametrize Problem (2.7) in γ to obtain:

$$f^{\text{sub}}(\gamma) = \max_{\substack{\mathbf{x} \in \mathcal{X} \\ \mathbf{y} \in \{0,1\}^n}} \left\{ \sum_{i=1}^n (\pi_i - \gamma) w_i y_i \cdot \prod_{k \in A(i)} \delta_k^{x_k} \mid x_i \leq y_i \ \forall i \in N \right\}.$$

Following the strategy of Section 2.3, we seek γ such that $f^{\text{sub}}(\gamma) = w_0 \gamma$. We define value functions at each vertex to consider its subtree's contribution to the parametrized problem, because we can still split decisions over the two children of each non-leaf vertex. In particular, we can add the new constraints to $V_\gamma(k)$ to

obtain a new value function $V_\gamma^{\text{sub}}(k)$:

$$V_\gamma^{\text{sub}}(k) = \max_{\substack{\mathbf{x} \in \mathcal{X}_k, \\ \mathbf{y} \in \{0,1\}^{|L(k)|}}} \left\{ \sum_{i \in L(k)} (\pi_i - \gamma) w_i y_i \cdot \prod_{k' \in A(i)} \delta_{k'}^{x_{k'}} \mid x_i \leq y_i \ \forall i \in L(k) \right\}.$$

If k is a leaf, then $x_k = 1$ implies $y_k = 1$ and the base case has the same value as before. The difference lies in deciding which products in $L(k)$ to offer online when none of the products in $L(k)$ are displayed. That is, $x_k = 0$ for $k \notin N$. Since $x_k = 0$ implies $x_i = 0$ for all $i \in L(k)$, y_i can be 0 or 1 for each leaf in the subtree. We take $y_i = 1$ if and only if $(\pi_i - \gamma) w_i \cdot \Delta_k$ is non-negative, so that product i only increases the value function. This occurs when $\pi_i \geq \gamma$. Let $(\cdot)^+$ denote $\max\{0, \cdot\}$. Applying this argument to the left and right subtree of k when k is not a leaf, we obtain the following dynamic program:

$$\begin{aligned} V_\gamma^{\text{sub}}(k) &= (\pi_k - \gamma) w_k \Delta_k & \forall k \in N, \\ V_\gamma^{\text{sub}}(k) &= \max \left\{ \begin{aligned} &V_\gamma^{\text{sub}}(\ell(k)) + V_\gamma^{\text{sub}}(r(k)), \\ &V_\gamma^{\text{sub}}(\ell(k)) + \Delta_k \cdot \sum_{i \in L(r(k))} (\pi_i - \gamma)^+ w_i, \\ &\Delta_k \cdot \sum_{i \in L(\ell(k))} (\pi_i - \gamma)^+ w_i + V_\gamma^{\text{sub}}(r(k)) \end{aligned} \right\} & \forall k \notin N. \end{aligned}$$

The parametrized problem takes value $f^{\text{sub}}(\gamma) = \max\{V_\gamma^{\text{sub}}(\text{root}), \sum_{i=1}^n (\pi_i - \gamma)^+ w_i\}$. For each γ , S^{PHY} is identified by reaching the base cases and $S^{\text{ON}} = S^{\text{PHY}} \cup \{i \mid \pi_i \geq \gamma\}$.

The number of operations to compute $V_\gamma^{\text{sub}}(\text{root})$ is the same as the original $V_\gamma(\text{root})$, but transforming the numerator and denominator of the objective function into linear functions to measure the number of iterations of Newton's method is more complicated. Simply applying Lemma 1 to the objective func-

tion in Problem (2.7) is not sufficient and results in:

$$\frac{\sum_{i=1}^n \pi_i w_i y_i + \sum_{k=1}^{2n-1} (\Delta_k - \Delta_{p(k)}) \cdot \sum_{i \in L(k)} \pi_i w_i x_k y_i}{w_0 + \sum_{i=1}^n w_i y_i + \sum_{k=1}^{2n-1} (\Delta_k - \Delta_{p(k)}) \cdot \sum_{i \in L(k)} w_i x_k y_i}.$$

The numerator and denominator of the objective function have quadratic terms $x_k y_i$. To turn the numerator and denominator into linear functions, we introduce variables z_{ki} for every vertex k and $i \in N$ such that $z_{ki} = x_k y_i$. The relationship can be rewritten as $z_{ki} \leq y_i$, $z_{ki} \leq x_k$, and $z_{ki} \geq x_k + y_i - 1$ to obtain the following mathematical program:

$$\max_{\substack{\mathbf{x} \in \mathcal{X} \\ \mathbf{y} \in \{0,1\}^n \\ \mathbf{z} \in \{0,1\}^{n(2n-1)}}} \left\{ \frac{\sum_{i=1}^n \pi_i w_i y_i + \sum_{k=1}^{2n-1} (\Delta_k - \Delta_{p(k)}) \cdot \sum_{i \in L(k)} \pi_i w_i z_{ki}}{w_0 + \sum_{i=1}^n w_i y_i + \sum_{k=1}^{2n-1} (\Delta_k - \Delta_{p(k)}) \cdot \sum_{i \in L(k)} w_i z_{ki}} \mid \begin{array}{l} x_i \leq y_i, \ z_{ki} \leq y_i, \\ z_{ki} \leq x_k, \ z_{ki} \geq x_k + y_i - 1 \end{array} \right\}.$$

The numerator and denominator of the objective function can be written as linear functions, but the feasible region becomes a subset of $\{0,1\}^{O(n^2)}$ and we have an upper-bound of $O(n^4 \log^2 n)$ iterations of Newton's method to find γ such that $f^{\text{sub}}(\gamma) = w_0 \gamma$. Hence we increase our runtime by a factor of $O(n^2)$ compared to Theorem 1.

2.6.2 Independent Online and In-store Assortments

In the second variation, we allow the online and in-store assortments to be independent of each other. Some stores offer in-store-only sales products that are not available online. The mathematical program is the same as Problem (2.7), but with the constraint $x_i \leq y_i$ removed. This recognizes that a product can be offered in-store ($x_i = 1$) but not online ($y_i = 0$). The parametrized problem also

drops this constraint, so the value function $V_\gamma^{\text{ind}}(k)$ is:

$$V_\gamma^{\text{ind}}(k) = \max_{\substack{\mathbf{x} \in \mathcal{X}_k, \\ \mathbf{y} \in \{0,1\}^{|L(k)|}}} \left\{ \sum_{i \in L(k)} (\pi_i - \gamma) w_i y_i \cdot \prod_{k' \in A(i)} \delta_{k'}^{x_{k'}} \right\}.$$

If k is not a leaf, then the dynamic program follows the structure of $V_\gamma^{\text{sub}}(k)$ because we choose the online assortment in the subtree that is not offered in-store immediately and postpone the decision in the subtree that offers at least one product in-store. The difference lies in the leaves because we can set $y_i = 0$ even when $x_i = 1$. Hence, we set $y_i = 1$ when $(\pi_i - \gamma) w_i \Delta_i$ is non-negative, so that we offer product i online when its per-unit revenue is at least the online expected revenue:

$$\begin{aligned} V_\gamma^{\text{ind}}(k) &= (\pi_k - \gamma)^+ w_k \Delta_k & \forall k \in N, \\ V_\gamma^{\text{ind}}(k) &= \max \left\{ \begin{aligned} &V_\gamma^{\text{ind}}(\ell(k)) + V_\gamma^{\text{ind}}(r(k)), \\ &V_\gamma^{\text{ind}}(\ell(k)) + \Delta_k \cdot \sum_{i \in L(r(k))} (\pi_i - \gamma)^+ w_i, \\ &\Delta_k \cdot \sum_{i \in L(\ell(k))} (\pi_i - \gamma)^+ w_i + V_\gamma^{\text{ind}}(r(k)) \end{aligned} \right\} & \forall k \notin N. \end{aligned}$$

The parametrized problem takes value $f^{\text{ind}}(\gamma) = \max\{V_\gamma^{\text{ind}}(\text{root}), \sum_{i=1}^n (\pi_i - \gamma)^+ w_i\}$. For each γ , S^{PHY} is identified by reaching the base cases and $S^{\text{ON}} = \{i \mid \pi_i \geq \gamma\}$. The runtime analysis is the same as Subsection 2.6.1 where we require $S^{\text{PHY}} \subseteq S^{\text{ON}}$. The number of operations to compute $V_\gamma^{\text{sub}}(\text{root})$ and $V_\gamma^{\text{ind}}(\text{root})$ are the same, and we have the same objective function.

2.7 Numerical Study: Modeling Power of Features Tree Model

Dzyabura and Jagabathula (2017) had demonstrated empirical evidence supporting their choice model, but it is NP-hard to compute the optimal assortment

even under the showroom setting. We show that we can use a features tree to approximate their model.

We use a generalization of Dzyabura and Jagabathula (2017)’s model as the ground-truth model. We test two aspects of our features tree model: its ability to approximate the true purchase probabilities and its ability to select an assortment that obtains a high percentage of the true optimal expected revenue. We also test against a third model, the benchmark model, to measure the performance of our features tree model against a simple, non-features-based choice model in the omnichannel retail setting. We run our tests for the showroom setting to focus on online customers.

We describe the ground-truth and benchmark models in this section but defer the mathematical details to Appendix A.3. For consistent notation across the three models, we use set notation S to describe the assortment being displayed in-store. Under the features tree model, the probability that a customer chooses product i is $P_i^T(S)$ and the expected revenue is $\Pi^T(S)$.

2.7.1 Ground-Truth and Benchmark Models

The ground-truth model that we test against is a generalization of Dzyabura and Jagabathula (2017)’s model. Suppose there are L feature classes and K feature values per class. Each product is created by a combination of one feature value per feature class. However, we do not require that a product exists for every combination, so that $n < K^L$ is possible. The ground-truth model allows any two products to share feature values over multiple classes.

In the ground-truth model, product i has initial preference weight w_i . A feature is denoted by its class-value pair, (ℓ, k) , where $1 \leq \ell \leq L$ and $1 \leq k \leq K$, and is associated with a multiplier $\delta_{\ell,k}$. Similar to the features tree model, a feature is seen when at least one of the products with that feature is displayed in-store. If feature (ℓ, k) is seen and is a feature of product i , then we update the preference weight of product i by multiplying w_i by $\delta_{\ell,k}$. After updating preference weights, customers make their purchase decisions from the full assortment N and the no-purchase option according to MNL. Given a display assortment S , the probability that customers choose product i is denoted $P_i^G(S)$ and the expected revenue is denoted $\Pi^G(S)$ under the ground-truth model.

As a benchmark model, we consider a simple extension of MNL to the omnichannel retail setting by dropping the features dependence between products. Each product's preference weight depends only on whether it is displayed in-store. In this case, product i has preference weight v_i if it is seen in-store and w_i otherwise. Customers then make their purchase decisions from the full assortment N and the no-purchase option according to MNL by applying the appropriate preference weights. Given a display assortment S , the probability that customers choose product i is denoted $P_i^B(S)$ and the expected revenue is denoted $\Pi^B(S)$ under the benchmark model.

2.7.2 Test of Predictive Ability

First, we test the features tree model's ability to approximate the true purchase probabilities if customers make purchase decisions according to the ground-truth model. To test the predictive ability of our features tree model, we gener-

ate many instances of the ground-truth model. For each instance, we generate purchase data by randomly selecting display assortments and the resulting purchase under the ground-truth model. We fit a features tree to each instance, and for a second set of display assortments, measure the difference between purchase probabilities under the features tree model and the ground-truth model.

Setup

An instance of the ground-truth model consists of L feature classes and K feature values per class, where $L \in \{2, 3\}$ and $K \in \{4, 5, 6\}$. The retailer offers $n = 15$ products, which are selected uniformly at random out of the K^L possible combinations of features. The features of product i is denoted $A(i)$. Each feature is associated with a utility value, and a product's utility is the sum of its features' utilities. Feature (ℓ, k) has utility $\mu_{\ell,k}$, such that $e^{\mu_{\ell,k}}$ is generated uniformly over $[0, 2]$. Following the interpretation of preference weights in MNL, the preference weight of product i is the exponential of its utility, such that $w_i = e^{\sum_{(\ell,k) \in A(i)} \mu_{\ell,k}}$. Finally, each feature (ℓ, k) is also associated with a multiplier $\delta_{\ell,k}$, which is generated uniformly over $[0.1, 1.9]$. The preference weight of the no-purchase option w_0 is chosen so that a customer who sees the full assortment N on display will walk out empty-handed with probability 0.1.

To obtain data from an instance of the ground-truth model, we generate $D = 2500$ assortments to display in-store. For each $d = 1, \dots, D$, an assortment S_d is generated from a binomial distribution such that each product is displayed with probability 0.1. From each assortment S_d , we generate the customer's purchase decision i_d , which is either the no-purchase option or one of the n products, according to the purchase probabilities under the ground-truth model.

The structure of the features tree depends on how we order the features. We start with $L!$ features trees, where the layers of each tree corresponds to a permutation of the L feature classes. The parameters of each features tree are computed by solving the features tree's corresponding maximum likelihood estimator. As the maximum likelihood estimator might not be concave, we use Ipopt (Wächter and Biegler (2006)) to find a local optimal solution. We choose the features tree with the highest log-likelihood when we substitute its estimated parameters back into its log-likelihood function. Similarly, we estimate the parameters of the benchmark model by finding a local optimal solution to its maximum likelihood estimator.

To test the performance of the features tree model and the benchmark model, we generate another $D = 2500$ assortments. The new assortments S'_d are generated in the same manner as the previous set of assortments, for $d = 1, \dots, D$. We test the performance of the features tree model by measuring the mean absolute error, MAE^T , between the purchase probability of each product under the ground-truth model and the features tree model. Similarly, we denote the mean absolute error of the benchmark model as MAE^B :

$$\text{MAE}^T = \frac{1}{Dn} \sum_{d=1}^D \sum_{i=1}^n |P_i^T(S'_d) - P_i^G(S'_d)|, \text{ and } \text{MAE}^B = \frac{1}{Dn} \sum_{d=1}^D \sum_{i=1}^n |P_i^B(S'_d) - P_i^G(S'_d)|.$$

In addition to measuring the performance of the features tree model against the ground-truth model, we also compare the performance of the features tree model against the benchmark model. We measure the reduction in prediction error as:

$$\text{ReduceError} = \frac{\text{MAE}^B - \text{MAE}^T}{\text{MAE}^B} \times 100\%.$$

Results

For each $L \in \{2, 3\}$ and $K \in \{4, 5, 6\}$, we generate $M = 100$ instances of the ground-truth model and fit a features tree model and a benchmark model according to the procedure above. For each instance $m = 1, \dots, M$, we compute MAE_m^T , MAE_m^B , and ReduceError_m . We report the average of these measures, as well as the 5th and the 95th percentile, in Table 2.1.

Overall, our features tree model performs well in estimating the purchase probabilities of the ground-truth model. On average, we make an error of 0.0114 when estimating the purchase probability of a product. At the 95th percentile, our MAE increases to 0.0189. The features tree model significantly outperforms the benchmark model, by reducing the error in estimating purchase probabilities by 22% on average.

In Figure 2.2, we present the results for $L \in \{2, 3\}$ and $K = 5$ by plotting MAE_m^T against MAE_m^B for $m = 1, \dots, M$. The plots for the other (L, K) pairs look similar and are not presented. The diagonal $y = x$ has been plotted as a dashed line. Most of the points lie above the diagonal, which implies that on an instance-by-instance basis, the benchmark model incurs a higher mean absolute error than the features tree model. When the benchmark model outperforms the features tree model, its mean absolute error is only slightly smaller than the mean absolute error of the features tree model. In contrast, there are some instances in which the mean absolute error of the features tree model is significantly smaller than the mean absolute error of the benchmark model.

L	K	Tree MAE ^T			Benchmark MAE ^B			ReduceError		
		5th perc.	95th perc.	avg.	5th perc.	95th perc.	avg.	5th perc.	95th perc.	avg.
2	4	0.0065	0.0156	0.0105	0.0085	0.0201	0.0141	1.61%	52.82%	24.53%
2	5	0.0058	0.0170	0.0105	0.0082	0.0208	0.0138	3.27%	44.89%	22.89%
2	6	0.0059	0.0160	0.0093	0.0076	0.0192	0.0123	4.05%	42.93%	22.76%
3	4	0.0082	0.0215	0.0132	0.0102	0.0265	0.0168	2.99%	42.17%	20.53%
3	5	0.0076	0.0213	0.0127	0.0098	0.0268	0.0164	4.43%	55.89%	21.33%
3	6	0.0070	0.0220	0.0123	0.0099	0.0260	0.0154	2.91%	46.77%	20.13%

Table 2.1: Mean Absolute Error in Estimating Purchase Probability for Features Tree and Benchmark Models

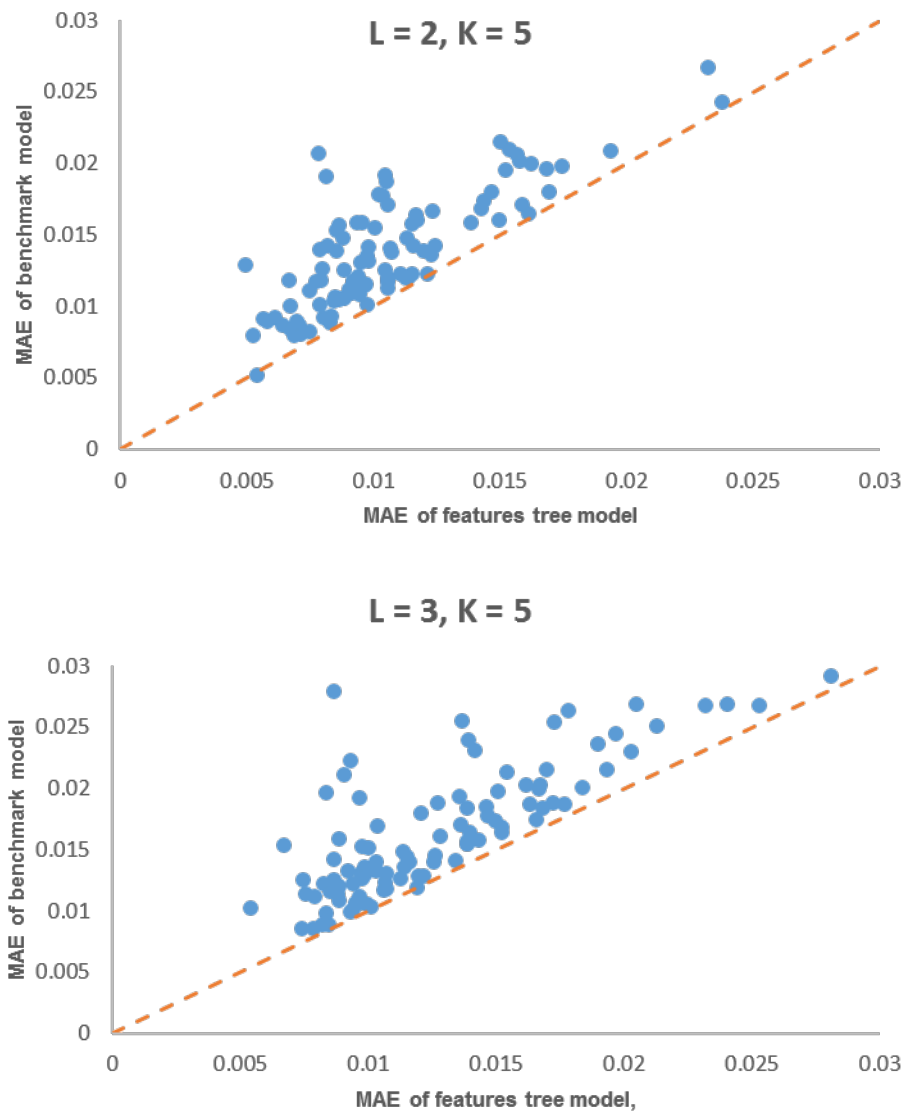


Figure 2.2: Comparison of the mean absolute error when using the features tree model versus the benchmark-model to approximate the ground-truth model.

2.7.3 Test of Assortments Selected by Features Tree

The main advantage of the features tree model over the ground-truth model is its computation tractability. Hence we want to know how the optimal assortment obtained from a fitted features tree would perform against the true optimal assortment.

Setup

For each instance of the ground-truth model in the previous subsection, we generate $J = 250$ scenarios by generating a new set of product prices for each scenario. The price of each product is generated uniformly at random over $[1, 10]$. As the set of prices change for each scenario, we denote the expected revenue function in scenario j as $\Pi_j^G(\cdot)$, because we are concerned with the expected revenue of each assortment under the ground-truth model.

For scenario $j = 1, \dots, J$, we compute the optimal assortment S_j^T under the features tree model, using the fitted features tree that was selected in the earlier test of predictive ability. We also compute the optimal solution S_j^B under the fitted benchmark model and the true optimal solution S_j^* by enumeration. We compare the expected revenue under the ground-truth model for each of the assortments S_j^T , S_j^B , and S_j^* . For each instance, we measure the percentage of the true optimal expected revenue earned by assortments S_j^T and S_j^B in scenarios $j = 1, \dots, J$, and then average over its J scenarios. We denote this measure by Earn^T and Earn^B respectively:

$$\text{Earn}^T = \frac{1}{J} \sum_{j=1}^J \frac{\Pi_j^G(S_j^T)}{\Pi_j^G(S_j^*)} \times 100\%, \text{ and } \text{Earn}^B = \frac{1}{J} \sum_{j=1}^J \frac{\Pi_j^G(S_j^B)}{\Pi_j^G(S_j^*)} \times 100\%.$$

We also measure the improvement that we obtain from using the features tree model over the benchmark model over all J scenarios of that instance:

$$\text{ImproveRev} = \frac{1}{J} \sum_{j=1}^J \frac{\Pi_j^G(S_j^T) - \Pi_j^G(S_j^B)}{\Pi_j^G(S_j^B)} \times 100\%.$$

Results

We reuse the $M = 100$ instances of the ground-truth model from the previous subsection, as well as the fitted features tree and benchmark models. For each instance $m = 1, \dots, M$, we computed the value of Earn_m^T , Earn_m^B , and ImproveRev_m , where each instance averages the performance of its J scenarios. We report the average of these measures, as well as the 5th and the 95th percentile, in Table 2.2.

Our features tree model performs quite well in choosing display assortments which capture a high percentage of the optimal expected revenue under the ground-truth model. On average, we capture more than 94% of the true optimal expected revenue. On the low end of performance at the 5th percentile, we capture 89% of the optimal expected revenue on average. Furthermore, we outperform the benchmark model by an average increase of 4% in expected revenue.

In Figure 2.3, we present the results for $L \in \{2, 3\}$ and $K = 5$ by plotting the values of Earn_m^T against the values of Earn_m^B for $m = 1, \dots, M$. The plots for the other (L, K) pairs look similar and are not presented. The diagonal $y = x$ has been plotted as a dashed line. Most of the points in the plots lie below the diagonal, which implies that on an instance-by-instance basis, the features tree model earns a higher expected revenue than the benchmark model when

L	K	Tree Earn ^T			Benchmark Earn ^B			ImproveRev		
		5th perc.	95th perc.	avg.	5th perc.	95th perc.	avg.	5th perc.	95th perc.	avg.
2	4	92.20%	98.28%	95.87%	87.40%	97.01%	92.49%	0.63%	10.26%	4.00%
2	5	89.68%	98.31%	95.36%	85.94%	96.49%	92.33%	0.80%	8.32%	3.58%
2	6	92.53%	98.49%	96.09%	88.14%	97.09%	93.30%	1.11%	7.20%	3.22%
3	3	87.31%	97.86%	93.95%	81.47%	96.26%	90.28%	0.47%	11.86%	4.58%
3	4	87.84%	96.92%	93.53%	82.79%	95.80%	89.96%	0.24%	9.96%	4.47%
3	5	86.97%	96.97%	93.24%	80.89%	94.46%	89.50%	0.60%	14.30%	4.82%
3	6	89.24%	97.72%	93.97%	85.25%	95.41%	90.85%	0.60%	9.19%	3.85%

Table 2.2: Percentage of Optimal Expected Revenue Earned by Using Features Tree and Benchmark Models

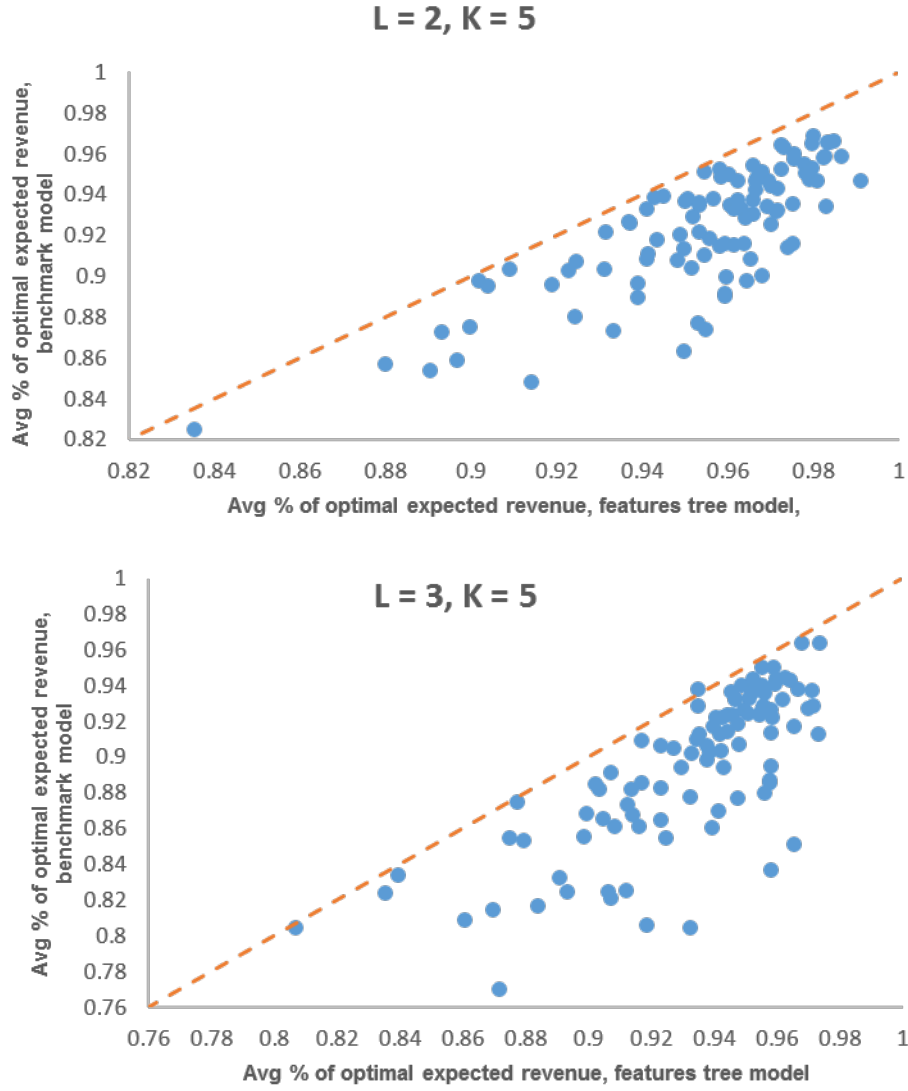


Figure 2.3: Average percentage of optimal expected revenue obtained when using the features tree model versus the benchmark model to compute the optimal assortment.

averaged over the J scenarios of that instance. Even when the benchmark model outperforms the features tree model, it is by a very small margin.

To conclude this section, our tests show that the features tree model is able to approximate the ground-truth model quite well, both in terms of estimat-

ing purchase probabilities and selecting the optimal assortment to display in-store. The ground-truth model is a generalization of Dzyabura and Jagabathula (2017)’s model and hence computationally difficult to optimize over. Our features tree model is a good substitute in that it is computationally tractable while achieving a 94% of the optimal expected revenue on average.

2.8 Numerical Study: Performance of FPTAS

We test our FPTAS to assess its practical performance compared to the its theoretical guarantee. We generate many instances of the OCA problem under a features tree model, and compare the expected revenue earned by our FPTAS against the upper-bound described in Section 2.5.

2.8.1 Setup

We assume that offline and online customers have the same preference weight for product i if they both see it. In other words, $\hat{v}_i = w_i \cdot \prod_{k \in A(i)} \delta_k$ and $\hat{v}_0 = w_0$. This focuses our test on the omnichannel aspect so that offline and online customers only differ in terms of what they are willing to purchase.

For each instance, we generate 32 products. Each product’s price π_i is generated uniformly over $[1, 10]$. Each product’s initial preference weight w_i is generated uniformly over $[1, 5]$.

Recall from Assumption 1 that our features tree T is a binary tree. Suppose vertex k is not a leaf nor the root, and feature k has multiplier $\delta_k = 1$. Then we can

contract the edge between k and its parent on T to obtain an equivalent features tree that is not binary. We use this idea to construct a general features tree. First, we set up a balanced binary tree with 32 products as leaves. For a leaf $i \in N$, a product-specific multiplier δ_i is generated uniformly over $[0.1, 1.9]$. For each of the 31 non-leaf vertices, its feature multiplier δ_k is set to 1 with probability β and generated uniformly over $[0.1, 1.9]$ with probability $1 - \beta$. Here, β is a parameter we vary in $\{0, 0.05, 0.1, 0.2\}$.

The no-purchase option is chosen so that a customer who sees the full assortment will refuse to purchase with probability $\eta \in \{0.05, 0.1\}$. Hence we set $w_0 = \hat{v}_0 = \eta \cdot \sum_{i=1}^n \hat{v}_i / (1 - \eta)$.

For each instance, we vary the fraction of online customers q . For each value of $q \in \{0.2, 0.4, 0.6, 0.8\}$, let x^q denote the assortment returned by our FPTAS with a performance guarantee of $\epsilon = 1/2$. Let $\Pi^q(\cdot)$ denote the expected revenue function and $\bar{\Pi}^q$ denote the upper-bound from Section 2.5. We measure the percentage of the upper-bound earned by our FPTAS solution:

$$\text{EarnLP}^q = \frac{\Pi^q(x^q)}{\bar{\Pi}^q} \times 100\%.$$

In our code, we used the constrained longest path formulation in Online Appendix A.2 to take advantage of the $O(n^2/\epsilon^2)$ improvement in runtime.

2.8.2 Results

For each combination of $(\eta, \beta) \in \{0.05, 0.1\} \times \{0, 0.05, 0.1, 0.2\}$, we generate $M = 50$ instances of the problem. For each instance m , we compute EarnLP_m^q and report the results in Table 2.3. Each row describes a combination of $(\eta, \beta) \in \{0.05, 0.1\} \times$

$\{0, 0.05, 0.1, 0.2\}$. The proceeding four blocks summarize the average, minimum, and maximum of EarnLP_m^q for varying values of $q \in \{0.2, 0.4, 0.6, 0.8\}$. Our code computes the FPTAS assortment for all values of q at the same time, so only one average runtime is reported in each row. We run our FPTAS with a performance guarantee of $\epsilon = 1/2$, but make our grid 30 times finer ($\epsilon = 1/60$) when we construct the upper-bound.

Our FPTAS achieves at least 96% of the upper-bound on average, even though we ran our FPTAS with a performance guarantee of $1/2$. The upper-bound uses a linear programming relaxation of our OCA problem and is not tight. In the worst case above, our FPTAS achieves 92.9% of the upper-bound. The average runtime of our FPTAS is 5.82 minutes, which translates to 349 seconds of CPU time.

2.9 Conclusion

We considered the assortment optimization problem of an omnichannel retailer, who must consider how his in-store assortment affects customers' preferences when they purchase from the full assortment in his online store. Our features tree is a simple and intuitive method to group products by features; this structure allowed us to develop a FPTAS to compute an approximately optimal assortment in the general setting via dynamic programming. We considered two extensions where the retailer chooses both his online and in-store assortments, and showed that they can be solved efficiently via a small modification to our dynamic program.

We fitted our features tree model to data simulated from a generalized

η	β	$q = 0.2$			$q = 0.4$			$q = 0.6$			$q = 0.8$			runtime (min)	
		avg.	min.	max.	avg.	min.	max.	avg.	min.	max.	avg.	min.	max.		
5%	0%	97.3%	93.0%	99.3%	98.3%	96.0%	99.4%	98.9%	97.1%	99.7%	99.5%	96.4%	99.8%	6.67	
5%	5%	97.1%	93.7%	99.2%	98.2%	96.3%	99.4%	98.9%	97.3%	99.7%	99.6%	97.5%	99.9%	6.99	
5%	10%	97.3%	92.9%	99.4%	98.2%	95.1%	99.5%	98.9%	97.2%	99.7%	99.6%	98.8%	99.9%	6.95	
5%	20%	96.9%	93.6%	99.2%	98.2%	95.8%	99.5%	98.9%	97.0%	99.6%	99.5%	98.5%	99.9%	6.54	
10%	0%	96.8%	93.2%	99.3%	97.9%	95.9%	99.6%	98.9%	97.7%	99.7%	99.6%	99.0%	99.9%	5.11	
10%	5%	96.8%	93.9%	99.2%	98.1%	94.9%	99.2%	99.1%	96.7%	99.7%	99.6%	98.5%	99.9%	4.81	
10%	10%	97.1%	93.5%	99.3%	98.2%	95.8%	99.3%	99.0%	97.5%	99.6%	99.7%	99.1%	99.9%	4.73	
10%	20%	96.8%	93.3%	99.2%	98.1%	96.6%	99.4%	99.0%	97.4%	99.7%	99.5%	96.1%	99.9%	4.74	

Table 2.3: Average, worst-case, and best-case performance of FPTAS on 50 instances of OCA under features tree model with $\epsilon = 1/2$. Time reported is for computing the FPTAS solution.

model. We showed that the fitted tree was able to approximate the true purchase probabilities of products with a mean absolute error of 0.01 and select assortments achieving 94% of the optimal expected revenue on average. When we ran our FPTAS over numerical examples with a performance guarantee of $1/2$ of the optimal expected revenue, we achieved 96% of the optimal expected revenue on average.

One future research direction is pricing in the omnichannel environment. Some retailers post different prices online and in-store, or offer promotions over one channel and not the other. Although previous papers have considered pricing for one product, pricing has not been considered in terms of assortments. Rather than simply using prices to modify customers' utilities, as is standard in MNL literature, perhaps an OCA pricing problem should also modify the fraction of customers purchasing online or in-store based on the average prices in the channels. This would represent the omnichannel environment more accurately, where customers are not loyal to one channel.

Assortment optimization in the omnichannel environment is quite new, and there could be better ways to explain how customers update preferences when they can research on either or both channels. Our model assumed that customers gather information from both channels and update preferences strictly based on the features tree. More general ways to update product preferences can be examined. Other choice models, like the nested logit model or the Markov chain model, could also be modified for the omnichannel setting.

Finally, parameter estimation is an important aspect of assortment optimization since representing customers' preferences accurately is a critical step before solving for the optimal assortment. In our numerical experiments, we used the

local optimal solutions to the maximum likelihood estimators. A future research direction is to obtain real-world data from an omnichannel retailer and test our model and estimator on the data to evaluate the usefulness of the features tree model in practice.

CHAPTER 3

**ASSORTMENT OPTIMIZATION UNDER THE MULTINOMIAL LOGIT
MODEL WITH PRODUCT SYNERGIES**

3.1 Introduction

Modeling customers' purchasing behaviours is critical in retail operations because it affects the retailer's decisions on what to sell in order to maximize his profit. Early work in inventory management assumed that demands are independent of the assortment being offered. Later, choice models in revenue management literature recognized that product demand might decrease if customers have more options. However, most choice models do not allow for synergistic effects. Synergy can increase the demand for a product when it is seen with some other products.

We study the assortment optimization problem under a synergistic version of the multinomial logit model (MNL). Customers associate a preference weight with each product, and the preference weight can increase via synergy. Synergy occurs between pairs of products when the retailer offers both of them in his assortment, even when customers purchase at most one product. The purchase probabilities of either product may increase or decrease, depending on the increase in preference weight and the weights of the other product. Marketing research shows that retailers can increase demand by offering a less attractive product to highlight the target product. The assortment optimization problem is to select a subset of products to offer, in order to maximize the retailer's expected profit.

Our Contributions

We consider a retailer who has access to n products. Each product has a base preference weight, which describes a customer’s preference for the product when it is seen alone. Each pair of products, i and j , also have a pair of synergistic preference weights, which describes how synergy from product i acts on j and vice versa. The purchase probability of a product is proportional to its preference weight in the offered assortment.

We use a graph to depict pairs of products with positive synergy when they are seen together. The assortment optimization problem under synergistic MNL is NP-hard for general synergy graphs. We restrict the product pairs with synergy and study the special cases of a synergy path and a synergy tree. We present algorithms that find the optimal assortment via dynamic programming, with runtime polynomial in the number of products. We extend our dynamic program to consider synergy graphs with low treewidth. In the case of the synergy path, we present a linear program that can recover the optimal assortment.

Literature Review

Our paper is inspired by the welfare-based choice models introduced by Feng et al. (2015). The authors introduced a welfare function, which takes as input the utilities of an assortment and outputs its welfare as a real number. Choice probabilities are given by the gradients of the welfare functions with respect to each product. The welfare function has three properties. Monotonicity ensures that welfare increases when all the utilities increase. Translation invariance states that purchase probabilities do not change if all utilities increased by the same

amount. Convexity ensures that a high-utility product improves welfare more than multiple low-utility products.

Feng et al. (2015) defined operations on welfare-based choice models which create new welfare-based choice models. One resulting model is the basis for our synergistic MNL. In their model, the total increase in preference weight from synergy (if any) between two products is equal to a weighted geometric mean of their base preference weights. In our model, we allow synergy weights to be any non-negative values. They focused on defining a new class of models, and did not discuss assortment optimization. We focus on the latter problem.

Evidence of synergy exists in marketing literature. Product sales can increase when an inferior product is introduced into the assortment (Simonson (1999)). In one study, a retailer selling a bread maker introduced a second, over-priced bread maker to his store. The demand for the original bread maker increased even though the assortment became larger. Products can also trigger a change in preference; Hanks et al. (2012) found that cookies sales increased in a cafeteria if applesauce was offered but not if green beans were offered.

A traditional approach to choice modeling is via utility-maximization. In the class of random utility models, a customer's utility is the sum of the mean utility plus a random noise. MNL (Luce (1959), McFadden (1973)) is the most famous model in this class, and the random noise for each product follows an independent standard Gumbel distribution. The preference weight of a product is the exponential of its mean utility, and the purchase probability of a product is proportional to its preference weight in the offered assortment.

MNL has the independence of irrelevant alternative (IIA) property: the ratio

of two products' purchase probabilities is unchanged regardless of what other products are offered. Our synergistic model follows MNL such that purchase probabilities are proportional to preference weights, but it is not subject to IIA because synergy can increase the weights. We discuss other extensions to MNL, and details can be found in Train (2003).

In the nested logit model, products are partitioned into nests by similarities. A customer chooses a nest, and then chooses a product within her chosen nest. The retailer can increase the probability that a customer chooses a nest by adding products to the nest, but that might not increase the purchase probability of individual products. In fact, incorporating synergy effect into the nested logit model might sacrifice its utility-maximizing property. The nested logit model has since been extended into the d-level nested logit model (Li and Huh (2011)) and to the generalized extreme value model (Train (2003)), where products belong to multiple nests.

In the mixed logit model (MMNL) studied by Bront et al. (2009) and Rusmevichientong et al. (2014), different customer types have different sets of preference weights. McFadden and Train (2000) showed that MMNL can approximate any random utility choice model. Désir et al. (2014) gave a fully-polynomial time approximation scheme to compute a $(1 - \epsilon)$ -optimal solution.

The random noises of utilities do not have to follow the Gumbel distribution. In the probit model first described by Thurstone (1927), the noises follow a multivariate normal distribution and the model does not exhibit the IIA property. Choice models outside the class of random utility models include the Markov Chain choice model (Blanchet et al. (2016)) and the non-parametric choice model. Paul et al. (2016) studied a special case of the non-parametric

choice model, where customers consider at most two products.

Our underlying parametric problem requires maximizing a quadratic function subject to binary variables. Unconstrained quadratic binary programming is NP-hard, and can be rewritten as an integer program. This problem can be classified by its underlying graph, where vertex i represents variable x_i and edge (i, j) exists if the coefficient of the quadratic term $x_i x_j$ is non-zero. Padberg (1989) studied the graph and the related integer program, and identified cases where the convex hull of the feasible region is integral. We use this graph to specify cases of synergistic MNL which can be solved efficiently. A survey of quadratic binary programming can be found in Kochenberger et al. (2014).

Our linear program for finding the optimal assortment under a synergy path is similar to the sales-based linear program (SBLP) described by Gallego et al. (2014). They studied a generalization of the network revenue management problem under MNL, where the retailer offers assortments over time subject to resource constraints. We use a similar strategy by taking the dual of the appropriate dynamic program in linear program form.

Organization

In Section 3.2, we describe synergistic MNL and its parametrized form, as well as the synergy graph. In Section 3.3, we focus on the synergy path and synergy tree, and use dynamic programming to solve the parametrized problem. In Section 3.4, we present a linear program which recovers the optimal assortment when we have a synergy path. We conclude in Section 3.5.

3.2 The Model

We describe the general model and the parametrized problem, and show that the general model is NP-hard.

3.2.1 Model and Notations

The retailer has access to n products, denoted $N = \{1, \dots, n\}$. Product i generates a profit of π_i when it is purchased by a customer. We do not require $\pi_i > 0$ for all products, only that at least one product has positive profit. The retailer can offer negative-profit products if they boost the preferences on high-profit products. The retailer chooses an assortment from N to sell in his store, which we denote by a binary vector $\mathbf{x} \in \{0, 1\}^n$ such that $x_i = 1$ if product i is in the assortment and 0 otherwise.

We describe customers' preferences by preference weights, and purchase probabilities are proportional to these weights. The preference weight of product i is a function of the assortment \mathbf{x} because synergy from the assortment can increase its preference weight. Product i has a base preference weight of $u_i \geq 0$, when it is the only product in the assortment. When product j is offered in the assortment, synergy between products i and j increases the preference weight of product i additively. We denote the synergy effect of product j on i by $v_i^j \geq 0$. Simultaneously, product i increases the preference weight of product j by v_j^i . Given an assortment \mathbf{x} , the preference weight of product i when it is offered is $u_i + \sum_{j \neq i} v_i^j x_j$.

Upon seeing assortment \mathbf{x} , a customer makes her purchase decision accord-

ing to MNL. If product i is offered, then her probability of purchasing product i is proportional to the preference weight of product i in the assortment, along with the no-purchase option. The no-purchase option is her ability to leave the store without a purchase, and we scale the preference weight of the no-purchase option to 1 without loss of generality. Hence, if assortment \mathbf{x} is offered, the probability that she buys product i is:

$$P_i(\mathbf{x}) = \frac{(u_i + \sum_{j \neq i} v_{ij}^j x_j) x_i}{1 + \sum_{k=1}^n (u_k + \sum_{j \neq k} v_{kj}^j x_j) x_k}.$$

The customer purchases at most one product, and synergy simply makes products look more attractive when they are offered together. The retailer's problem is to find an assortment \mathbf{x} that maximizes his expected profit: $\Pi(\mathbf{x}) = \sum_{i=1}^n \pi_i P_i(\mathbf{x})$.

For cleaner notation, we define $\bar{\pi}_{i,j}$ as the weighted average of the profits of products i and j , and $v_{i,j}$ as the total increase to the preference weight of the assortment due to synergy when both products i and j are offered, with the convention that $i < j$. When $v_{ij}^i = v_{ij}^j = 0$ so that there is no synergy in either directions, then $\bar{\pi}_{i,j} = 0$ and $v_{i,j} = 0$. Otherwise, define:

$$\bar{\pi}_{i,j} = \frac{\pi_i v_{ij}^j + \pi_j v_{ij}^i}{v_{ij}^j + v_{ij}^i}, \quad \text{and} \quad v_{i,j} = v_{ij}^j + v_{ij}^i.$$

Using this notation, our assortment optimization problem is $\max_{\mathbf{x} \in \{0,1\}^n} \Pi(\mathbf{x})$, which expands out to:

$$\max_{\mathbf{x} \in \{0,1\}^n} \frac{\sum_{i=1}^n \pi_i u_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^n \bar{\pi}_{i,j} v_{i,j} x_i x_j}{1 + \sum_{i=1}^n u_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^n v_{i,j} x_i x_j}. \quad (3.1)$$

3.2.2 The Parametrized Problem

We apply a standard parametrization technique for fractional combinatorial problems (Radzik (1998)). Suppose there exists an assortment \mathbf{x} with expected profit greater or equal to δ . By rearranging $\Pi(\mathbf{x}) \geq \delta$, we observe:

$$\sum_{i=1}^n (\pi_i - \delta) u_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^n (\bar{\pi}_{i,j} - \delta) v_{i,j} x_i x_j \geq \delta.$$

We can maximize the left side over all assortments and the inequality would still hold. The parametrized problem, with value $h(\delta)$, is:

$$h(\delta) = \max_{\mathbf{x} \in \{0,1\}^n} \sum_{i=1}^n (\pi_i - \delta) u_i x_i + \sum_{i=1}^{n-1} \sum_{j=i+1}^n (\bar{\pi}_{i,j} - \delta) v_{i,j} x_i x_j. \quad (3.2)$$

Claim 2. *Given $h(\delta)$ as defined above, let δ^* be the optimal expected profit of our assortment optimization problem. Then the following are true: i) $h(\delta) > \delta$ if $\delta < \delta^*$, ii) $h(\delta) < \delta$ if $\delta > \delta^*$, and iii) $h(\delta) = \delta$ if $\delta = \delta^*$.*

Suppose we can solve Problem (3.2) with corresponding value $h(\delta)$ for any $\delta \geq 0$. Since $h(0) > 0$ and $h(\delta)$ is monotone decreasing to $-\infty$, one method to find δ^* from Claim 2 is via Newton's method. By using the techniques that transform a quadratic binary program to an integer program (Padberg (1989)), we can show that Newton's method finds δ^* in $O(n^4 \log^2 n)$ iterations of computing $h(\delta)$. Hence, we can solve Problem (3.1) in polynomial time if we can solve Problem (3.2) in polynomial time.

Unfortunately, Problem (3.2) is a special case of quadratic binary programming, which is NP-hard in general (Kochenberger et al. (2014)). Problem (3.1) is also NP-hard, and we prove the next theorem via a reduction from the maximum independent set problem.

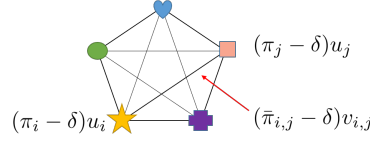


Figure 3.1: Synergy graph on five products - Synergy depicted by an edge between a pair of products.

Theorem 3. *The assortment optimization problem under synergistic MNL is NP-hard.*

Instead, one way to classify quadratic binary programs and identify cases which are solvable in polynomial-time is to consider the underlying graph (Padberg (1989)). Construct a graph $G = (V, E)$ such that vertex i represents variable x_i and an edge goes from vertex i to j if the coefficient of the quadratic term $x_i x_j$ is non-zero. In Problem (3.2), the coefficient $(\bar{\pi}_{i,j} - \delta)v_{i,j}$ changes as we vary δ and we want to consider the quadratic binary program for all possible values of δ , so we add an edge (i, j) whenever $v_{i,j} > 0$. Hence $V = N$ and $E = \{(i, j) : v_{i,j} > 0\}$. We call this graph the synergy graph (e.g. Figure 3.1).

If our synergy graph has at least two components, then Problem (3.2) can be broken up into smaller sub-problems containing only the products in the component. Without loss of generality, we assume our synergy graph is connected. In practice, synergy does not exist between arbitrary pairs of products. We focus on the cases where the synergy graph is a path or a tree and extend our results to consider synergy graphs with low treewidth.

3.3 Optimal Assortments on Synergy Paths and Trees

We present dynamic programs to solve Problem (3.2) when the synergy graph is a path, a tree, or has low treewidth.

3.3.1 A Synergy Path

Suppose the synergy graph is a path. For example, consider ice cream, which improves in taste as the fat content increases. The presence of a compromise ice cream, which balances the fat content and taste, can increase demands for a better-tasting, fattier ice cream and a healthier, less tasty ice cream (Simonson (1999)).

Since the synergy graph is a path, we can order the products so that product i creates synergy with products $i - 1$ and $i + 1$ only. Let $\bar{\pi}_i = \bar{\pi}_{i,i+1}$ and $v_i = v_{i,i+1}$ for $i = 1, \dots, n - 1$ when we consider the synergy path. Problem (3.2) simplifies to:

$$h(\delta) = \max_{\mathbf{x} \in \{0,1\}^n} \sum_{i=1}^n (\pi_i - \delta) u_i x_i + \sum_{i=1}^{n-1} (\bar{\pi}_i - \delta) v_i x_i x_{i+1}.$$

We use a dynamic program to compute the value of $h(\delta)$. For $i \geq 2$, define the value function $V_i(x_{i-1})$ to be the maximum value that products i to n can contribute to the objective function of Problem (3.2), given the state x_{i-1} for product $i - 1$. At product 1, there is no previous product whose state we have to consider. We attribute the synergy effect between products $i - 1$ and i to product i , given the decision x_{i-1} . Hence, our value function can be written as the following, with $h(\delta) = V_1$:

$$\begin{aligned} V_1 &= \max_{x_1, \dots, x_n} \sum_{i=1}^n (\pi_i - \delta) u_i x_i + \sum_{i=1}^{n-1} (\bar{\pi}_i - \delta) v_i x_i x_{i+1}, \\ V_i(x_{i-1}) &= \max_{x_i, \dots, x_n} (\bar{\pi}_{i-1} - \delta) v_{i-1} x_{i-1} x_i + \\ &\quad \sum_{j=i}^n (\pi_j - \delta) u_j x_j + \sum_{j=i}^{n-1} (\bar{\pi}_j - \delta) v_j x_j x_{j+1}. \end{aligned}$$

The presence or absence of product $i - 1$ does not affect products $i + 1$ to n once we have decided whether or not to offer product i . If product i is offered, then

its contribution is $R_i^\delta(x_{i-1}) := (\bar{\pi}_{i-1} - \delta)v_{i-1}x_{i-1} + (\pi_i - \delta)u_i$. The first term is the synergy contribution between products $i-1$ and i , given the decision of offering product $i-1$. The second term is the base contribution of product i . We can rewrite our value functions using the definition of $R_i^\delta(x_{i-1})$ and $V_{i+1}(x_i)$ to get our dynamic program:

$$\begin{aligned} V_1 &= \max_{x_1 \in \{0,1\}} (\pi_1 - \delta)x_1 + V_2(x_1), \\ V_i(x_{i-1}) &= \max_{x_i \in \{0,1\}} R_i^\delta(x_{i-1}) \cdot x_i + V_{i+1}(x_i), \quad \forall i = 2, \dots, n, \\ V_{n+1}(x_n) &= 0. \end{aligned} \tag{3.3}$$

Our base cases are $V_{n+1}(0) = V_{n+1}(1) = 0$, and we compute the dynamic program backwards from product n to 1. If we decide $x_i = 0$, then $V_i(x_{i-1}) = V_{i+1}(0)$, and we immediately lose the synergy effect with both products $i-1$ and $i+1$, as well as the base value $(\pi_i - \delta)u_i$ from product i . If we decide $x_i = 1$, then $V_i(x_{i-1}) = R_i^\delta(x_{i-1}) + V_{i+1}(1)$. We get the base value from product i , and we may get the synergy effect with product $i-1$, depending on the value of x_{i-1} . Furthermore, we have the opportunity to create synergy with product $i+1$ at the next value function.

We conclude with the runtime analysis. For a fixed δ , we can compute $h(\delta)$ in $O(n)$ operations because there n products, and $O(1)$ states and $O(1)$ decisions at each state per product. We could run Newton's method to find δ^* . Alternatively, we can find δ^* by solving a linear program in Section 3.4, which has $O(n)$ variables and constraints. This allows us to compute the optimal assortment with $O(n)$ operations plus one linear programming computation.

3.3.2 A Synergy Tree

Suppose the synergy graph is a tree. For example, an inexpensive, generic-brand product creates synergy with the entry-level products of several national brands. In turn, the entry-level product of each national brand creates synergy with the higher-quality products of its brand.

Given a product i , let p_i represent its parent and C_i represent the set of its children. To be consistent with our earlier notation, we index the products so that the indices satisfy $p_i < i < c$ for $c \in C_i$. If product i is a leaf of the synergy tree, then $C_i = \emptyset$. Also, let T_i denote the set of vertices in the subtree rooted at product i . Since a product only creates synergy with either its parent or its children, we can rewrite Problem (3.2) as:

$$h(\delta) = \max_{\mathbf{x} \in \{0,1\}^n} \sum_{i=1}^n (\pi_i - \delta) u_i x_i + \sum_{i=1}^n \sum_{c \in C_i} (\bar{\pi}_{i,c} - \delta) v_{i,c} x_i x_c.$$

Suppose we are considering whether or not to offer product i , and we already know whether its parent is offered (i.e. x_{p_i}). Then the decision of whether or not to offer product i is independent of the decisions for the ancestors of p_i , as well as the other children of p_i . We can focus on the subtree rooted at i . When product i is not the root, we define our value function $V_i(x_{p_i})$ as the maximum contribution from vertices in T_i to the objective function of Problem (3.2), given our decision on offering product p_i . We attribute the synergy between products p_i and i to product i and the value function $V_i(x_{p_i})$. The value function at the root, V_{root} , has no synergy from a parent and is by definition equal to $h(\delta)$. Our

value functions are:

$$\begin{aligned}
V_{\text{root}} &= \max_{x_1, \dots, x_n} \sum_{i=1}^n (\pi_i - \delta) u_i x_i \\
&\quad + \sum_{i=1}^n \sum_{c \in C_i} (\bar{\pi}_{i,c} - \delta) v_{i,c} x_i x_c, \\
V_i(x_{p_i}) &= \max_{x_j: j \in T_i} (\bar{\pi}_{p_i,i} - \delta) v_{p_i,i} x_{p_i} x_i \\
&\quad + \sum_{j \in T_i} (\pi_j - \delta) u_j x_j \\
&\quad + \sum_{j \in T_i} \sum_{c \in C_j} (\bar{\pi}_{j,c} - \delta) v_{j,c} x_j x_c.
\end{aligned}$$

We apply the strategy from the case of the synergy path. Suppose product i is offered in our assortment and it is not the root. Its contribution to the objective function of Problem (3.2) is the synergy contribution with its parent given the decision x_{p_i} and its base contribution, $R_i^\delta(x_{p_i}) := (\bar{\pi}_{p_i,i} - \delta) v_{p_i,i} x_{p_i} + (\pi_i - \delta) u_i$. If product i is the root, then its contribution is $(\pi_{\text{root}} - \delta) u_{\text{root}}$. We can rewrite the value functions as a dynamic program, using the definition of $R_i^\delta(x_{p_i})$ and $V_c(x_i)$ for $c \in C_i$:

$$\begin{aligned}
V_{\text{root}} &= \max_{x_{\text{root}} \in \{0,1\}} (\pi_{\text{root}} - \delta) u_{\text{root}} x_{\text{root}} + \sum_{c \in C_{\text{root}}} V_c(x_{\text{root}}), \\
V_i(x_{p_i}) &= \max_{x_i \in \{0,1\}} R_i^\delta(x_{p_i}) \cdot x_i + \sum_{c \in C_i} V_c(x_i), \quad \forall i \neq \text{root}.
\end{aligned} \tag{3.4}$$

We solve the dynamic program starting from the leaves until we reach the root. If product i is a leaf, then $C_i = \emptyset$ and $V_i(x_{p_i}) = \max_{x_i \in \{0,1\}} R_i^\delta(x_{p_i}) \cdot x_i$, so the base case looks similar to the synergy path's base case. If product i is not a leaf, then $x_i = 0$ implies that $V_i(x_{p_i}) = \sum_{c \in C_i} V_c(0)$. This means we lose the base contribution from product i and the synergy with its parent, as well as any chance of synergy with its children. If $x_i = 1$, then we keep the base contribution of product i and possibly synergy with its parent. The decision of whether we create synergy

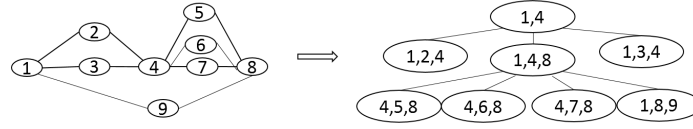


Figure 3.2: Tree decomposition - On the left, we have a synergy graph on 9 products. On the right, we have a tree decomposition of the synergy graph. A vertex on the tree represents a subset of vertices on the original graph.

with each child is deferred to the corresponding child.

We need to compute the value function at each vertex of our synergy tree, and there are $O(1)$ states and $O(1)$ decisions at each state per product. Hence, we can solve the dynamic problem in Problem (3.4) and compute $h(\delta) = V_{\text{root}}$ in $O(n)$ operations for any fixed δ . We can find δ^* using Newton's method, or we can find δ^* via a linear program similar to the one in Section 3.4. The number of operations to compute the optimal assortment when we have a synergy tree or a synergy path is on the same order.

The dynamic program that we construct for the synergy tree suggests that we can use tree decompositions to consider synergy graphs with low treewidth. Tree decompositions are a method to represent graphs with a tree such that each tree vertex represents a subset of vertices on the original graph, and each graph vertex is associated with a subtree on the tree. A graph can have many tree decompositions, and each decomposition is associated with a measure called width. The treewidth of a graph is the minimum width over all of its tree decompositions. Tree-based dynamic programs can be modified into efficient algorithms when the graph has low treewidth (Williamson and Shmoys (2011)). Details of this extension are deferred to Appendix B.1, and an example of a tree decomposition is depicted in Figure 3.2.

Theorem 4. *For a synergy graph G , suppose we are given a tree decomposition with*

width t and $O(n)$ vertices. Then it takes $O(2^t n)$ operations to solve Problem (3.2). If we use Newton's algorithm to find δ^* , then the total operations needed to compute the optimal assortment is $O(2^t n^5 \log^2 n)$.

For example, if the synergy graph has no K_4 minor so that it is a series-parallel graph, then it has treewidth $t^* = 2$ (Wald and Colbourn (1983), Brandstadt et al. (1999)). In our theorem, we assume that a tree decomposition of low width is given. If a graph has treewidth t^* , algorithms exist to find a tree decomposition with width $t = t^*$ and at most $O(n)$ vertices (Bodlaender (1996)), with runtime polynomial in n but exponential in t^* . On the other hand, there exists algorithms to find a tree decomposition with width $t = O(t^* \log n)$ in polynomial runtime (Bodlaender et al. (1995)), which increases the runtime of the dynamic program by a factor of n compared to using a tree decomposition with minimal width.

3.4 Sales-Based Linear Program for Synergy Path

We revisit our synergy path, and present a linear program that immediately reveals the optimal assortment. This allows a practitioner to take advantage of linear program solvers and avoids solving Problem (3.2) repeatedly.

Suppose the value of δ is fixed. The values of V_1 and $V_i(x_{i-1})$ are not constrained to be integers. If we simply want to compute the value of $h(\delta) = V_1$, then we can transform our dynamic program into a linear program by replacing the decision process at each state with two constraints: the value of $V_i(x_{i-1})$ is greater or equal to the outcomes of both decisions $x_i = 0$ and $x_i = 1$. We transform our dynamic program in Problem (3.3) into the following linear program

with variables V_1 and V_i^x , where $x \in \{0, 1\}$ represents the state x_{i-1} for $i \geq 2$. By construction, the optimal objective value of this linear program is $h(\delta)$.

$$\begin{aligned}
& \min V_1 & (3.5) \\
& \text{s.t. } V_1 \geq V_2^0 \\
& V_1 \geq (\pi_1 - \delta)u_1 + V_2^1 \\
& V_i^0 \geq V_{i+1}^0 & \forall i = 2, \dots, n \\
& V_i^0 \geq (\pi_i - \delta)u_i + V_{i+1}^1 & \forall i = 2, \dots, n \\
& V_i^1 \geq V_{i+1}^0 & \forall i = 2, \dots, n \\
& V_i^1 \geq (\bar{\pi}_{i-1} - \delta)v_{i-1} + (\pi_i - \delta)u_i + V_{i+1}^1 & \forall i = 2, \dots, n \\
& V_{n+1}^0 = V_{n+1}^1 = 0.
\end{aligned}$$

Let \mathcal{V} denote the set of constraints above. Claim 2 tells us that the linear program has optimal objective value δ^* if and only if we had created this linear program with $\delta = \delta^*$. Furthermore, the constraints in \mathcal{V} are linear in δ , so we can treat δ as a variable. Specifically, we add an extra constraint to the linear program, $\delta = V_1$, to obtain:

$$\min_{V \in \mathcal{V}, \delta} \{V_1 : \delta = V_1\}. \quad (3.6)$$

Lemma 3. *The optimal objective value of Problem (3.6) is equal to the optimal expected profit of Problem (3.1), δ^* .*

Lemma 3 lets us find δ^* by solving a linear program with $O(n)$ constraints and variables, and immediately proceed to computing $h(\delta^*)$. This avoids iterating over different values of δ in Newton's method and repeatedly solving the dynamic program. These techniques also work for finding δ^* for the synergy tree, and the linear program for a synergy tree also has $O(n)$ constraints and variables.

We can take one more step to find the optimal assortment directly via the solution of a linear program when we have a synergy path. Since Problem (3.6) is a feasible linear program with finite optimal objective value, strong duality guarantees that its dual is feasible and has optimal objective value δ^* . By taking the dual of Problem (3.6) and making the appropriate transformation of variables, we obtain:

$$\begin{aligned}
& \max \sum_{i=1}^n \pi_i y_i + \sum_{i=1}^{n-1} \bar{\pi}_i w_i & (3.7) \\
& \text{s.t. } y_i/u_i \geq w_i/v_i & \forall i = 1, \dots, n-1 \\
& y_{i+1}/u_{i+1} \geq w_i/v_i & \forall i = 1, \dots, n-1 \\
& y_0 \geq y_i/u_i + y_{i+1}/u_{i+1} - w_i/v_i & \forall i = 1, \dots, n-1 \\
& y_0 + \sum_{i=1}^n y_i + \sum_{i=1}^{n-1} w_i = 1 \\
& \mathbf{y}, \mathbf{w} \geq 0.
\end{aligned}$$

Problem (3.7) looks similar to the SBLP in Gallego et al. (2014), except for the terms related to synergy: w_i . If we can construct assortments from solutions to Problem (3.7), then we can interpret y_i as the probability that a customer purchases product i due to its base preference weight, and w_i as the additional probability that a customer purchases either products i or $i+1$ due to the synergy created when both are present. We are interested in solutions satisfying the following condition.

Condition 1. *We are interested in solutions (\mathbf{y}, \mathbf{w}) to Problem (3.7) of the following form:*

1. If $w_i = 0$, then $y_i = 0$ or $y_{i+1} = 0$.
2. If $w_i > 0$, then $y_i/u_i = y_{i+1}/u_{i+1} = w_i/v_i$.

3. If $y_i > 0$, then $y_i/u_i = y_0$.

Our goal is to show that extreme points of Problem (3.7) satisfies condition 1. Moreover, there is a one-to-one correspondence between extreme points and assortments. Lemma 4 maps assortments to solutions satisfying condition 1.

Lemma 4. *Given an assortment \mathbf{x} , there exists a solution (\mathbf{y}, \mathbf{w}) which is feasible to Problem (3.7) with objective value equal to $\Pi(\mathbf{x})$. Furthermore, (\mathbf{y}, \mathbf{w}) satisfies condition 1.*

We now consider the reverse direction, and show that solutions (\mathbf{y}, \mathbf{w}) that satisfy condition 1 map to assortments. Our main theorem proves that every extreme point solution satisfies condition 1.

Lemma 5. *Suppose (\mathbf{y}, \mathbf{w}) is feasible to Problem (3.7) and satisfies condition 1. If $x_i = \mathbb{1}[y_i > 0]$, then assortment \mathbf{x} has expected profit equal to the objective value of (\mathbf{y}, \mathbf{w}) .*

Theorem 5. *Every extreme point of the polytope in Problem (3.7) satisfies condition 1. Hence, given an extreme point optimal solution $(\mathbf{y}^*, \mathbf{w}^*)$ to Problem (3.7), we can recover an optimal assortment \mathbf{x}^* by taking $x_i^* = \mathbb{1}[y_i^* > 0]$.*

In summary, computing the optimal assortment of synergistic MNL when we have a synergy path can be achieved by solving a simple linear program. This allows us to avoid running the dynamic program altogether.

3.5 Conclusion

We presented a synergistic version of MNL for assortment optimization. Synergy is created if the preference weight of at least one product in a pair of prod-

ucts increases when both are present in the assortment. The optimal assortment can be computed efficiently when the synergy graph is a path or a tree. Furthermore, when we have a synergy path, the optimal assortment can be found by solving a simple linear program.

In terms of future research, it may be possible to extend our model to encompass cannibalization across products in addition to synergy. The natural approach would be to set the synergy weights $v_i^j < 0$, but doing so might violate the underlying assumptions that lets us parametrize our assortment optimization problem. A careful analysis would be needed to determine whether our lemmas and theorems are still valid.

Parameter estimation and validity of this model are also interesting future directions. The log-likelihood function of our model is not concave, so parameter estimation would have to rely on local optimal solutions if the maximum likelihood estimator is used. Under this limitation, it would be interesting to obtain real-world data and test whether synergistic MNL performs well with estimating purchase probabilities and computing optimal assortments.

Acknowledgment: We thank Jacob Feldman for his help in developing the proof of Theorem 3, and David Williamson for suggesting the use of graphs with low treewidth.

CHAPTER 4

ASSORTMENT OPTIMIZATION WITH DYNAMIC SUBSTITUTION AND INVENTORY STOCKING COSTS

4.1 Introduction

For many retailers, deciding what products to offer and how many units to stock are important and difficult questions. In the standard assortment optimization problem where we maximize expected revenue per customer, there is already a complex trade-off between offering many products to avoid customers leaving, and offering fewer products to avoid cannibalizing demand from products with high revenue. When we consider assortment optimization where the retailer has to decide on an assortment of products and the stocking levels of his inventory to satisfy customers who arrive over a selling horizon, the problem becomes even more complicated due to stock-out effects. If a customer's favourite product has stocked out, she may be willing to substitute to a different product based on the current assortment. This behaviour can increase demand for the substitute product beyond the retailer's expectation, and lead to cascading stock-out effects that affect the purchasing decisions of subsequent customers.

We study the joint assortment optimization and stocking problem of a retailer who has to decide what products to offer in his store and the corresponding inventory levels of the products in the assortment. The retailer has n products at his disposal. At the beginning of a finite selling horizon, the retailer chooses the products to offer to his customers and their corresponding stocking levels. The retailer pays a cost-per-unit to acquire his inventory.

Customers make purchasing decisions according to the non-parametric choice model. Each customer has a consideration set, which is the set of products that she is willing to purchase and is independent of the retailer's assortment. Her preference list ranks the products in her consideration set in the order of decreasing preferences. Customers arrive one-by-one and each customer purchases the highest-ranked product in her preference list which is still in-stock. In other words, if a preceding customer prefers product i to product j , but had to substitute to product j , then product i must have stocked-out and the current customer cannot purchase product i . In dynamic substitution, the retailer has no control on the assortment seen by each customer beyond choosing the initial stocking levels. He does not get to remove or replenish products to influence a customer's purchasing decision. For example, a clothing retailer cannot remove products from the shelves nor replenish inventory from his suppliers in the middle of the day. Hence, a customer's purchasing decision depends on the purchases of the preceding customers and the resulting stock-outs.

The assortment optimization problem with dynamic substitution and inventory stocking costs is to select an assortment of products and their corresponding stocking levels in order to maximize the retailer's expected profit. The expected profit is given by the expected revenue minus the stocking costs. Once the retailer has made his decisions, the stocking cost is deterministic. However, the revenue is stochastic and based on the preference lists of the customers and their order of arrival to the store.

Our Contributions

We study assortment optimization with dynamic substitution and inventory stocking costs. The underlying choice model which explains how customers make purchasing decision is the non-parametric, preference-ranking model. We consider the deterministic and stochastic setting of our problem over a finite time horizon of T periods. In the stochastic setting, the retailer knows a distribution on the number of customers arriving over the T periods, as well as a distribution on each customer's preferences. We refer to the outcome as the customers arrival sequence, which includes the information on the number of customers who arrive to the store, their preference lists, and the order in which they arrive. He decides on an assortment and stocking levels before the customers arrival sequence is revealed. His goal is to determine the assortment, as well as the stocking levels, which maximizes his expected profit. The stochastic setting is extremely difficult; given an assortment and stocking levels for the products in the assortment, computing the expected revenue would require solving a dynamic program with a state space which has size exponential in the number of products. As a result, we turn to studying the deterministic setting to gain insights and build a heuristic for the stochastic setting.

In the deterministic setting, the retailer knows the customers arrival sequence before he decides on the assortment and stocking levels. The preference lists of all of the customers are revealed in advance. Upon seeing the preferences of his customers, the retailer's problem is to select an assortment of products to offer in his store, as well as the stocking levels. He incurs a stocking cost for his inventory. Each customer obtains her most preferred product which is still in-stock upon her arrival to the store. The goal in the deterministic setting is to

maximizes the retailer's profit.

In the deterministic setting, we show that the problem of finding an optimal assortment with stocking levels is NP-hard, even when each customer considers and ranks at most two products. As a result, we restrict the structure of customers' preference lists. First, we study the case of K -choosy customers, which are customers who consider at most K products. This portrays the idea that customers are attached to their favourite products and are not willing to substitute beyond their K -th favourite product. We present a randomize algorithm which finds a solution that guarantees at least $\frac{2}{K} \cdot (1 - K)^{K-1}$ -fraction of the optimal profit. The algorithm works by carefully rounding the solution of the linear programming relaxation of an appropriate integer program. Second, we study customers with preference lists that are intervals over a central, objective ranking over the n products. This describes the case where products can be ranked objectively, such as products which increase in quality as their prices increase, or products which increase in size as the number of functions increase. Customers have constraints over the central ranking, such as maximum budget and size, or minimum quality and number of functions. For a fixed interval length K , we present an algorithm which computes the optimal assortment and stocking levels. The runtime of our algorithm is $O(nT^K)$, where T is the number of customers who arrive over the selling horizon.

In the stochastic setting, the retailer knows the distribution on the number and types of customers arriving to his store. He determines the assortment and stocking levels, and he incurs a deterministic stocking cost based on his decision. On the other hand, his revenue is a random quantity because he does not know how many customers will arrive nor their preference lists. The retailer

is interested in maximizing his expected revenue net of his stocking costs. We present a heuristic for the stochastic setting built upon our algorithms for the deterministic setting. Our heuristic samples customers arrival sequences and computes the optimal assortment for each sequence. It then chooses the assortment and stocking levels that earns the highest profit when averaged over all our samples. Via simulations, we show that our heuristic performs well.

Literature Review

There are two main components to our choice model: the underlying non-parametric, preference-ranking choice model of each arriving customer, and the multi-period nature of dynamic substitution.

When we focus on a single customer arrival, the non-parametric choice model associates a probability with the arrival of each customer type. A customer's type refers to her ranking of the products in her consideration set, also known as her preference list. The problem is to choose an assortment to maximize the expected revenue for one incoming customer. One issue with the non-parametric choice model is that customer types and arrival probabilities cannot be encoded in polynomial space. Instead, the literature focuses on the non-parametric choice model with a limited support on customer types.

Mahajan and Van Ryzin (2001*b*) introduced the use of the non-parametric choice into revenue management literature, but to the best of our knowledge, Honhon et al. (2012) were the first to consider assortment optimization under this choice model. They considered four structures on customers' preference lists and presented algorithms to compute the optimal assortment using short-

est path constructions and dynamic programming. Their one-way substitution model is what we refer to as customers with preference lists that form intervals over a central ranking. They also considered models which organize products as vertices on a tree. The root represents the most generic product and customers either substitute from specialized products to the generic product or in the reverse direction, depending on the model.

Another way to restrict the non-parametric model is to limit the size of consideration sets, which limits the number of different preference lists. Marketing literature shows that customers tend to consider a small number of options when they make a purchase. Aouad, Farias, Levi and Segev (2015) studied the assortment optimization problem where a customer makes her purchasing decisions according to the non-parametric choice model, with the additional restriction that a customer considers at most K products. They showed that it is NP-hard to approximate the optimal revenue within a factor of $O(K^{1-\epsilon})$ or $O(\log^{1-\epsilon} r_{\max}/r_{\min})$, where r_{\max}, r_{\min} are the highest and lowest prices respectively. They also showed that their second bound is tight and proved that it is achievable by a revenue-ordered assortment. When customers consider at most K products, they presented a ϵK -approximation algorithm. Paul et al. (2016) considered choosy customers who consider at most two products. They showed that a FPTAS for finding the optimal assortment does not exist unless $P = NP$ and presented a 2-approximation algorithm. In a subsequent work, Feldman et al. (2017) extended the result to K -choosy customers, and provided an algorithm which recovers $\frac{2}{K} \cdot (1 - K)^{K-1}$ -fraction of the optimal expected revenue. This improves the approximation guarantee of Aouad, Farias, Levi and Segev (2015). We use similar rounding techniques as Feldman et al. (2017) for the dynamic substitution problem and we obtain the same performance guarantee for

K -choosy customers.

In a related work, Aouad, Farias and Levi (2015) studied the consider-then-choose model. Instead of restricting the size of the consideration sets, they restricted the structure on the preference lists. Their model assumed that there is a central ranking of the products. Each customer has a consideration set of the products, and her preference list ranks the products in her consideration set according to the central ranking. Hence, the number of different preference lists, L , is equal to the number of different consideration sets. The assortment optimization problem under this model is still NP-hard, and they presented a graph decomposition which collapses the state space of the related dynamic program. Although the state space is still exponential when L is arbitrary, they showed that their dynamic program can be solved in a quadratic number of operations when $L = O(1)$ and a polynomial number of operations when $L = O(\log n)$. Furthermore, they presented efficient algorithms to compute the optimal assortment when preference lists take the form of intervals or laminar families over the central ranking. The latter case is similar to the tree-based models in the work of Honhon et al. (2012).

Other popular choice models in the literature include the class of random utility models, of which the multinomial logit (MNL) model is perhaps the most well-known. A comprehensive review of related random utility models is described by Train (2003). Also of current interest is the Markov Chain choice model described by Blanchet et al. (2016) and Feldman and Topaloglu (2014).

When the literature considers assortment optimization with customer arrivals over time, the retailer may or may not control the assortment seen by each customer. When the retailer has no control, the problem can be further

divided into static or dynamic substitution. In static substitution, customers make their purchasing decisions based on the initial assortment and ignore subsequent stock-outs. If their first-choice product under the initial assortment is unavailable, then they leave without a purchase. In dynamic substitution, customers modify their choices based on the products still in-stock. Hence their purchasing decisions are affected by the retailer's assortment and stocking decisions, as well as the purchases of the preceding customers.

Static models are reminiscent of catalog orders. Each customer makes her purchasing decision based on the same assortment of products, and is not aware of any stock-outs. If a customer decides to purchase a product and the product has stocked-out, then the sales is lost. Van Ryzin and Mahajan (1999) considered static substitution with MNL as the underlying choice model, such that all products have the same revenue and stocking cost. They showed that the optimal assortment is preference-weight-ordered, so that it is optimal to offer products with the k highest preference weights. After determining their optimal assortment, they used a newsvendor model to find the optimal stocking levels. Around the same time, Smith and Agrawal (2000) presented a dynamic substitution model, but required that any product offered in the assortment should not stock-out with a high probability. This ensures that the total demand of products offered are not dependent on the stocking quantity and their model is closer to a static substitution model.

Dynamic substitutions are representative of retailers who operate in physical stores and online. Unlike catalog orders, each customer is aware that certain products have stocked-out when she makes a purchasing decision, and she chooses amongst the assortment of products that have not stocked-out. Most

retailers offer the same assortment to customers until products stock-out. This is true even for online retailing when we consider major retailers like Gap and Target. To the best of our knowledge, the first paper to consider dynamic substitution was by Mahajan and Van Ryzin (2001*b*). The authors showed that the profit earned by a fixed product satisfies useful properties, but the total profit function over all products is not concave in the initial stocking levels. They also consider the continuous setting where customers can be satisfied with fractional quantities of products, such as when we consider liquid products. Even in the continuous setting, the profit function is not quasi-concave in the initial stocking levels and may have local optima. Under the assumption that the products are continuous rather than discrete, the authors presented a sample path gradient method to compute the stocking levels. They also presented two heuristic based on the newsvendor model. Unlike their paper, we only consider discrete unit of products being sold.

In a follow up paper, Mahajan and Van Ryzin (2001*a*) considered firm competition with dynamic substitution, so that customers substitute by visiting another retailer if they cannot obtain the product at their favourite store. This is a game theoretical approach and the authors showed that a pure Nash equilibrium exists under regularity conditions. At equilibrium, the products are overstocked. We do not take this approach and consider only one retailer selling all the products.

Recently, there has been a renewal of interest in dynamic substitutions, most notably by Goyal et al. (2016), Aouad, Levi and Segev (2015*a*) and Aouad, Levi and Segev (2015*b*). Each of these papers considered products without stocking costs. Instead there is a cardinality constraint on the total units of inventory,

which we call a shelf-space constraint. As Goyal et al. (2016) noted, this is reasonable when costs for products are already sunk, perhaps because the products are already in back-storage, and space is the primary constraint.

Goyal et al. (2016) showed that the dynamic assortment problem is NP-hard even when we maximize the expected revenue for a single choosy customer, whose preference list is unknown. This contrasts with our NP-hardness proof, which focuses on multiple customers arriving over time such that their preference lists are known to the retailer. They provided an approximation algorithm when the customers arrival sequence satisfies three assumptions. First, the number of customers follows a distribution with increasing failure rate. Second, customers have nested preference lists and rank products in order of increasing prices. Each customer considers the k least expensive products. Finally, they assumed access to an efficient oracle for evaluating the expected revenue given an assortment with initial stocking levels. Goyal et al. (2016) provided a PTAS for finding an optimal assortment and stocking levels.

Aouad, Levi and Segev (2015a) studied dynamic substitution with general customer choice models where the number of customers follows a distribution with an increasing failure rate. Their algorithm guarantees $1/4 \cdot (1 - 1/e)$ of the optimal expected revenue if all products have the same price and a $O(\log(r_{\max}/r_{\min}))$ -approximation otherwise. The performance guarantee can be improved to $O(\log \log(r_{\max}/r_{\min}))$ when preference lists are intervals of a central ranking. When customers have nested preference lists, their algorithm guarantees $1 - 1/e$ of the optimal expected revenue without any restrictions on the distribution of the number of customers.

Aouad, Levi and Segev (2015b) also assumed that the number of customers

follow a distribution with increasing failure rate. Each customer makes a purchasing decision according to MNL. They introduced the notions of restricted submodularity and restricted monotonicity, which allows them to build an approximation algorithm via greedy approaches and the multi-product newsvendor problem. In order to use the greedy algorithm, they also showed how one can estimate the expected revenue given an initial inventory via Monte Carlo simulation, even when certain products have low preference weights and their purchases are considered rare events.

The aforementioned papers in dynamic substitution over time are distinct from ours in two ways. First, we consider non-refundable stocking costs instead of the shelf-space constraint. This would be relevant for a retailer that is in the process of ordering products. Second, the papers above consider the stochastic problem directly whereas we approach the problem from the deterministic setting. We show that the problem is NP-hard even when the customers arrival sequence is known in advance.

Organization

In Section 4.2, we present our model and notation. In Section 4.3, we show that the problem of computing the optimal assortment with stocking levels is NP-hard, even when the customers arrival sequence is revealed to the retailer. We turn to the special cases of K -choosy customers and customers whose preference lists are intervals on a central ranking. We present a $\frac{2}{K} \cdot (1 - K)^{K-1}$ -approximation in the former case, and a polynomial time algorithm to compute the optimal assortment and stocking levels in the latter case. In Section 4.4, we study the stochastic setting and present a heuristic. We also present a linear program

to upper-bound the average profit over a sample set of customers arrival sequences. This upper-bound allows us to measure the performance of the solution returned by our heuristic. Our approximation algorithm for K -choosy customers and the heuristic for the stochastic setting are tested in Section 4.5. Finally, we conclude in Section 4.6.

4.2 The Model

We first describe the model in the context of the deterministic setting. The retailer has access to n products, denoted by $N = \{1, \dots, n\}$, which he can choose to offer in his store. A unit of product i costs the retailer c_i to stock, and the stocking cost is incurred regardless of whether the product is later sold. If a unit of product i is sold, then the retailer earns a revenue of r_i for a net profit of π_i .

Our selling horizon consists of T periods, with one customer arriving in each time period. We refer to the customer who arrives into the store at time period t as customer t . At the beginning of the selling horizon, each customer states her preference list over the products in her consideration set. Her consideration set is a subset of N , and these are the products that she is willing to purchase. Customer t only ranks the products in her consideration set, and we refer to the resulting preference ranking as her preference list σ_t . We indicate customer t preferring product i over j by $i \succ_t j$. We abuse notation slightly and use $i \in \sigma_t$ to indicate that product i is in customer t 's consideration set. We may also say that product i is in her preference list. The customer leaves the store without a purchase if none of the products in σ_t are available to her.

Upon observing σ_t for $t = 1, \dots, T$, the retailer decides on the number of

units of product i to offer, u_i , for $i \in N$. If $u_i = 0$, then product i is not in the assortment. Customers then make their purchasing decision in order of their arrivals. A customer t purchases the first product in σ_t that has not stocked-out. To define dynamic substitution formally, we use a binary vector \mathbf{x}_t to state the purchasing decision of customer t . Customer t purchases product i if and only if $x_{t,i} = 1$. Dynamic substitution is captured by the following definition:

Definition 1. *Given T customers and u_i units of product i for $i \in N$, we say that customer t purchases product $i \in \sigma_t$ ($x_{t,i} = 1$) by substituting dynamically if:*

1. *For all products $j \succ_t i$, all units of product j have been sold before time t ($\sum_{t'=1}^{t-1} x_{t',j} = u_j$).*
2. *There is positive units of product i remaining at time t ($\sum_{t'=1}^{t-1} x_{t',i} < u_i$).*

The second condition can be simplified by ensuring that we never sell more units of product i than the retailer has stocked. Our assortment optimization problem can be written as the following integer program:

$$\begin{aligned}
& \max \sum_{i=1}^n \sum_{t=1}^T r_i x_{t,i} - \sum_{i=1}^n c_i u_i & (4.1) \\
& s.t. \sum_{i=1}^n x_{t,i} \leq 1 & \forall t = 1, \dots, T \\
& \sum_{t=1}^T x_{t,i} \leq u_i & \forall i = 1, \dots, n \\
& u_j - \sum_{t'=1}^{t-1} x_{t',j} \leq T \cdot (1 - x_{t,i}) & \forall t = 1, \dots, T; i, j \in \sigma_t : j \succ_t i \\
& x_{t,i} = 0 & \forall t = 1, \dots, T; i \notin \sigma_t \\
& \mathbf{x}_t \in \{0, 1\}^n & \forall t = 1, \dots, T \\
& \mathbf{u} \geq 0, \mathbf{u} \in \mathbb{Z}^n.
\end{aligned}$$

The first constraint tells us that each customer purchases at most one product. The second constraint tells us that the retailer cannot sell more units than he has stocked, and ensures that the second condition of dynamic substitution is satisfied. The third constraint says that a customer cannot purchase product i if she prefers product j and product j has not stocked out. This constraint ensures that the first condition of dynamic substitution is satisfied. Finally, the fourth condition states that a customer only purchases a product in her preference list. Note that even though $x_{t,i}$ are binary variables in Problem (4.1), their values are uniquely determined by the decisions of \mathbf{u} .

The above formulation does not carry directly over to the stochastic setting. When customers arrival sequences are stochastic, it is not clear how we can efficiently compute the expected profit of a given \mathbf{u} . Using a dynamic program to compute the expected profit would require a state space which is exponential in the number of products. In Section 4.4, we consider how Problem (4.1) can be modified to consider the expected profit over a sample of customers arrival sequences.

4.3 Dynamic Substitution in the Deterministic Setting

In this section, we prove that the assortment optimization with dynamic substitution under the deterministic setting is NP-hard. Instead of solving for the stocking levels \mathbf{u} directly, we introduce an alternative view of our problem where the retailer can decide to discontinue products at each time period. Under the deterministic setting, there is a mapping between product discontinuation times and stocking levels. This strategy also allows us to study the determinis-

tic setting when we restrict the structure of customers' preference lists. First, we consider the case where customers are K -choosy, so that each customer considers and ranks at most K products. Next, we consider customers with preference lists which are intervals on a central ranking.

4.3.1 NP-hardness of the Setting

One simplifying observation we can make in the deterministic setting is that it is never optimal to overstock. Given any solution \mathbf{u} , we can determine in polynomial time the purchasing decision of each customer $t = 1, \dots, T$. Any units of product i that are not sold could have been removed, which results in a lower stocking cost without changing revenue. Thus, we can restate the objective function of Problem (4.1) in terms of the profit, $\pi_i = r_i - c_i$, because every unit of product i is sold at optimality. Without loss of generality, we can rewrite Problem (4.1) as:

$$\begin{aligned}
& \max \sum_{i=1}^n \pi_i u_i \\
& s.t. \sum_{i=1}^n x_{t,i} \leq 1 & \forall t = 1, \dots, T \\
& \sum_{t=1}^T x_{t,i} = u_i & \forall i = 1, \dots, n \\
& u_j - \sum_{t'=1}^{t-1} x_{t',j} \leq T \cdot (1 - x_{t,i}) & \forall t = 1, \dots, T; i, j \in \sigma_t : j \succ_t i \\
& x_{t,i} = 0 & \forall t = 1, \dots, T; i \notin \sigma_t \\
& \mathbf{x}_t \in \{0, 1\}^n & \forall t = 1, \dots, T \\
& \mathbf{u} \geq 0, \mathbf{u} \in \mathbb{Z}^n.
\end{aligned}$$

We prove that our problem is NP-hard, and construct our proof for the simplified deterministic setting. The proof is deferred to Appendix C.1.

Theorem 6. *The assortment optimization problem with dynamic substitution and deterministic customer arrivals is NP-hard.*

We prove Theorem 6 via a reduction from the maximum independent set problem. Given a graph $G = (V, E)$ where $n = |V|$, $m = |E|$, and D is one larger than the maximum degree of the vertices in G , we construct an instance of our assortment optimization problem with $n + 1$ products and $n + m$ customers. We show that G has an independent set of size K if and only if the assortment optimization problem has a solution \mathbf{u} that achieves a profit of $nD + K$.

4.3.2 Product Discontinuation Times vs Stocking Levels

We present an alternative view of the problem by considering the optimal time to discontinue product i , τ_i , rather than its beginning stocking level. The basis for this view is that a customer does not care how many units of product i are on the shelves. Rather, she only cares about whether product i is still available. In this view, the retailer selects an assortment at the beginning of the selling horizon, and has the power to discontinue a product at any time without future replacement. Under the deterministic setting, we show that the problem of deciding how many units to stock is equivalent to the problem of deciding when to discontinue products. We restate our assortment optimization problem in terms of product discontinuation times. Then, we prove that the two problems are equivalent by showing how we can map solutions \mathbf{u} to τ and vice versa.

In the product discontinuation view, customer t purchases her highest-ranked product $i \in \sigma_t$ subject to $t \leq \tau_i$. If $\tau_i = 0$, then product i is not included in the assortment. In other words, only the products that have not been discontinued by the retailer are available to customer t . Notice that the purchasing decision of customer t is not affected by the purchasing decisions of the customers 1 through $t - 1$ who preceded her into the store, even though we later show that the two problems are equivalent in the deterministic setting. Mathematically, given a vector of product discontinuation times τ , we can compute customer t 's purchase as:

$$x_{t,i} = \mathbb{1}[i \in \sigma_t, t \leq \tau_i] \cdot \prod_{\substack{j: j \in \sigma_t \\ j > i}} \mathbb{1}[t > \tau_j].$$

We introduce a new integer program to model the product discontinuation problem. Let \mathbf{y}_t be a binary vector that tells us which products are available at time t ; that is, $y_{t,i} = \mathbb{1}[t \leq \tau_i]$. Equivalently, $\tau_i = \max\{t : y_{t,i} = 1\}$, because a product becomes unavailable ($y_{t,i} = 0$) after time τ_i . We consider the following product discontinuation problem:

$$\begin{aligned} \max \quad & \sum_{t=1}^T \sum_{i=1}^n \pi_i x_{t,i} & (4.2) \\ \text{s.t.} \quad & y_{t,i} \geq y_{t+1,i} & \forall t = 1, \dots, T-1; i = 1, \dots, n \\ & x_{t,i} \leq y_{t,i} & \forall t = 1, \dots, T; i = 1, \dots, n \\ & x_{t,i} + y_{t,j} \leq 1 & \forall t = 1, \dots, T; i, j \in \sigma_t : j > i \\ & x_{t,i} = 0 & \forall t = 1, \dots, T; i \notin \sigma_t \\ & \mathbf{x}_t, \mathbf{y}_t \in \{0, 1\}^n & \forall t = 1, \dots, T. \end{aligned}$$

The first constraint tells us that if product i is available for purchase at time $t + 1$, then it should also be available at time t . That is, the retailer cannot reverse

a decision to discontinue a product. The second constraint tells us that customer t can only purchase product i if it is available. The third constraint says that customer t cannot purchase product i when a product that is higher on her preference list is still available. She does not substitute to product i unless the retailer has discontinued all products that she prefers over product i . The fourth constraint again ensures that she only purchases products in her preference list.

We can drop the constraint that $\sum_{i=1}^n x_{t,i} \leq 1$ because this is a redundant constraint. Suppose by contradiction that $x_{t,j} = x_{t,i} = 1$ and $j >_t i$. Then $x_{t,i} + y_{t,j} \geq x_{t,i} + x_{t,j} = 2$. This violates the third constraint. Hence this formulation ensures that every customer purchases at most one product.

Lemma 6. *Problem (4.1) is equivalent to Problem (4.2).*

The proof is deferred to Appendix C.1. This alternative view, with the retailer deciding on the period t to discontinue each product, is used to construct our approximation for K -choosy customers and for customers with preference lists which are intervals over a central ranking.

4.3.3 K -Choosy Customers

Paul et al. (2016) defined a choosy customer to be a customer who considers at most two products before leaving the store. We use this terminology and say that a K -choosy customer is one who considers at most K products before leaving the store: $|\sigma_t| \leq K$.

We show that when customers are K -choosy, we can compute an assortment and subsequent product discontinuation times such that we are guaranteed to

achieve $\frac{2}{K} \cdot (1 - K)^{K-1}$ -fraction of the optimal profit. To do this, we first show that the basic optimal solutions to the linear programming relaxation of Problem (4.2) are half-integral. This means that if $(\mathbf{x}_t, \mathbf{y}_t)_{t=1, \dots, T}$ is a basic optimal solution, we have $(\mathbf{x}_t, \mathbf{y}_t)_{t=1, \dots, T} \in \{0, 1/2, 1\}^{2nT}$. Following this result, we show that we can round such a solution to a feasible solution of the integer program such that we achieve our approximation guarantee.

Lemma 7. *Suppose $(\mathbf{x}_t, \mathbf{y}_t)_{t=1, \dots, T}$ is a basic optimal solution to the linear programming relaxation of Problem (4.2). Then $(\mathbf{x}_t, \mathbf{y}_t)_{t=1, \dots, T}$ is half-integral.*

The proof of this has been relegated to Appendix C.1, and is similar to the one in Paul et al. (2016). Let Π^* denote the optimal objective value of Problem (4.2) and Π^{LP} denote the optimal objective value of its linear programming relaxation.

We construct an integral solution $(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t)_{t=1, \dots, T}$ from the optimal solution $(\mathbf{x}_t, \mathbf{y}_t)_{t=1, \dots, T}$ to the linear programming relaxation of Problem (4.2). The initial temptation would be to consider each non-integral variable independently, or to consider each time period t . However, this could easily violate the monotonicity constraint $y_{t,i} \geq y_{t+1,i}$. If we rounded $y_{t,i} = 1/2$ down to 0, then all subsequent $y_{t',i}$ must also be 0. Instead, for each $i \in N$, we want to round all $y_{t,i}$ in the same direction for $t = 1, \dots, T$. Fix a product i . If there exists t such that $y_{t,i} = 1/2$, then there must exist some time periods s_1, s_2 with $1 \leq s_1 \leq t$ and $t \leq s_2 \leq T$, such that $y_{t',i} = 1/2$ if and only if $s_1 \leq t' \leq s_2$. By rounding all $y_{t',i}$ during $s_1 \leq t' \leq s_2$ in the same direction, we would not violate the monotonicity constraint. We can then set $\bar{x}_{t,i} = \min\{\{\bar{y}_{t,i}\} \cup \{1 - \bar{y}_{t,j} : j \succ_t i\}\}$ to ensure that $(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t)_{t=1, \dots, T}$ is a feasible solution. This is summarized in Algorithm 3. This randomized algorithm can be derandomized via the standard method of conditional expectations (Spencer

(1987), Williamson and Shmoys (2011)).

Algorithm 3: Randomized algorithm to compute solution to Problem (4.2) which guarantees $\frac{2}{K} \cdot (1 - K)^{K-1}$ -fraction of the optimal profit, for K -choosy customers

Input : Optimal solution $(\mathbf{x}_t, \mathbf{y}_t)_{t=1, \dots, T}$ to the linear programming relaxation of Problem (4.2)

Output: Integral solution $(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t)_{t=1, \dots, T}$ to Problem (4.2)

Let Z_i be a random variable which takes value 1 with probability $1/K$ and 0 otherwise;

for $i = 1, \dots, n$ **do**

Determine realization z_i of Z_i ;

for $t = 1, \dots, T$ **do**

$\bar{y}_{t,i} = \mathbb{1}[y_{t,i} = 1] + \mathbb{1}[y_{t,i} = 1/2] \cdot \mathbb{1}[z_i = 1]$;

end

end

for $i = 1, \dots, n$ **do**

for $t = 1, \dots, T$ **do**

$\bar{x}_{t,i} = \min\{\{\bar{y}_{t,i}\} \cup \{1 - \bar{y}_{t,j} : j \succ_t i\}\}$;

end

end

Return $(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t)_{t=1, \dots, T}$.

Theorem 7. Suppose we construct a solution $(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t)_{t=1, \dots, T}$ from the half-integral optimal solution $(\mathbf{x}_t, \mathbf{y}_t)_{t=1, \dots, T}$ of the linear programming relaxation of Problem (4.2). Then $(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t)_{t=1, \dots, T}$ achieves at least $\frac{2}{K} \cdot (1 - K)^{K-1}$ -fraction of the optimal profit.

Proof. Suppose in Algorithm 3, we take Z_i to be a random variable which takes value 1 with probability $q \in (0, 1)$ and 0 otherwise. Linearity of expectation tells us that the expected profit under $(\bar{\mathbf{x}}_t, \bar{\mathbf{y}}_t)_{t=1, \dots, T}$ is:

$$E \left[\sum_{t=1}^T \sum_{i=1}^n \pi_i \bar{x}_{t,i} \right] = \sum_{t=1}^T \sum_{i=1}^n \pi_i E[\bar{x}_{t,i}].$$

We want to compare the value of $E[\bar{x}_{t,i}]$ to $x_{t,i}$. We show that $E[\bar{x}_{t,i}] \geq 2q(1 - q)^{K-1} x_{t,i}$, and that the lower-bound is maximized at $q = 1/K$.

If $x_{t,i} = 1$, then $y_{t,i} = 1$ and $y_{t,j} = 0$ for all $j \succ_t i$. Hence, $\bar{y}_{t,i} = 1$ and $\bar{y}_{t,j} = 0$ so that $\bar{x}_{t,i} = 1$. Else if $x_{t,i} = 0$, then either $y_{t,i} = 0$ or there exists $j \succ_t i$ such that $y_{t,j} = 1$. Hence either $\bar{y}_{t,i} = 0$ or there exists some j such that $\bar{y}_{t,j} = 1$, which implies $\bar{x}_{t,i} = 0$. In these two cases, we have $\bar{x}_{t,i} = x_{t,i}$. Finally, we consider the case where $x_{t,i} = 1/2$.

If $x_{t,i} = 1/2$, then $y_{t,i} \in \{1/2, 1\}$ and $y_{t,j} \in \{0, 1/2\}$ for all $j \succ_t i$. Notice $\bar{x}_{t,i} = 1$ if and only if $\bar{y}_{t,i} = 1$ and $\bar{y}_{t,j} = 0$ for all $j \succ_t i$, and 0 otherwise. From Algorithm 3, we have $Z_i = 1$ with probability q and $Z_j = 0$ with probability $(1 - q)$. Hence we have:

$$E[\bar{x}_{t,i}] \geq q \cdot (1 - q)^{K-1} = 2q \cdot (1 - q)^{K-1} x_{t,i}.$$

To obtain the inequality, observe that we only round the non-integral terms, with at most one term which needs to be rounded up. Furthermore, there are at most $K - 1$ products which precede product i in the preference list. We can check that $q \cdot (1 - q)^{K-1}$ is maximized at $q = 1/K$ for $q \in [0, 1]$.

Finally, to lower-bound the expected revenue of our integral solution:

$$\begin{aligned} E \left[\sum_{t=1}^T \sum_{i=1}^n \pi_i \bar{x}_{t,i} \right] &= \sum_{t=1}^T \sum_{i=1}^n \pi_i E[\bar{x}_{t,i}] \\ &\geq \sum_{t=1}^T \sum_{i=1}^n \pi_i \left(\frac{2}{K} \left(1 - \frac{1}{K} \right)^{K-1} x_{t,i} \right) \\ &= \frac{2}{K} \left(1 - \frac{1}{K} \right)^{K-1} \cdot \Pi^{\text{LP}} \\ &\geq \frac{2}{K} \left(1 - \frac{1}{K} \right)^{K-1} \cdot \Pi^*. \end{aligned}$$

When customers are 2-choosy, the solution above is a 2-approximation. We can recover the the associated stocking level by applying Lemma 6. Hence, we can find an assortment with stocking levels to guarantee $1/2$ of the optimal profit when customers are 2-choosy. \square

4.3.4 Customers with Interval Preference Lists

We turn to consider the setting where customers' preference lists are intervals over a central ranking. A central ranking can be interpreted as an objective ranking over the products. For example, we can consider winter jackets such that prices increase as quality increase, and customers may prefer less expensive jackets. Alternatively, the size of kitchen appliances can increase as the number of functions increases, and customers may prefer smaller appliances. Each customer has requirements on her purchase, such as minimum quality or number of functions, and maximum price or size.

Without loss of generality, we may assume that the central ranking is $1 \succ 2 \succ \dots \succ n$. For each customer t , there exists products $i_t \leq j_t$ such that $i_t \succ_t i_t + 1 \succ_t \dots \succ_t j_t$, and the other products are not in the customer's preference list. It is possible that $i_t = j_t$, which implies that customer t considers only one product. Let K be the maximum number of products that customers consider: $K = \max\{j_t - i_t + 1 : t = 1, \dots, T\}$.

By imposing an interval structure on K -choosy customers, we can design an algorithm that computes the optimal product discontinuation times. For a fixed K , this algorithm has runtime that is polynomial in n and T . The algorithm's runtime is exponential in K when K is arbitrary. For ease of presentation, we present our algorithm for the case of $K = 3$, and discuss how it can be extended for larger K .

Consider the case of $i \geq 3$. Suppose product i is in the preference list of customer t . There are at most three products, including product i itself, that can affect customer t 's decision on whether or not to purchase product i . In

particular, she purchases product i if $t \leq \tau_i$ and one of three conditions hold: $i)$ $i-1 \notin \sigma_t$, $ii)$ $i-1 \in \sigma_t, i-2 \notin \sigma_t$ and $t > \tau_{i-1}$, or $iii)$ $i-2 \in \sigma_t$ and $t > \max\{\tau_{i-2}, \tau_{i-1}\}$. In the first condition, customer t 's first-choice is product i . In the second condition, customer t 's first-choice is product $i-1$, but it has stocked-out. In the third condition, customer t prefers products $i-2$ and $i-1$ over product i , but both have stocked-out. Since $K = 3$, these are the only conditions under which customer t purchases product i . This implies that the only product discontinuation times which affect the total profit from selling product i are τ_{i-2} , τ_{i-1} , and τ_i .

Suppose we know τ_{i-2} , τ_{i-1} , and τ_i , and let $w_i(\tau_{i-2}, \tau_{i-1}, \tau_i)$ denote the total profit that we obtain from selling units of product i under these discontinuation times. This profit can be computed as:

$$w_i(\tau_{i-2}, \tau_{i-1}, \tau_i) = \pi_i \cdot \left(\begin{array}{l} \sum_{t=1}^T \mathbb{1}[i \in \sigma_t, i-1 \notin \sigma_t, t \leq \tau_i] \\ + \sum_{t=1}^T \mathbb{1}[i-1 \in \sigma_t, i-2 \notin \sigma_t, \tau_{i-1} < t \leq \tau_i] \\ + \sum_{t=1}^T \mathbb{1}[i-2 \in \sigma_t, \max\{\tau_{i-1}, \tau_{i-2}\} < t \leq \tau_i] \end{array} \right).$$

The first sum accounts for the customers whose first choice is product i . The second sum accounts for the customers whose first choice is product $i-1$ but substitute because product $i-1$ has been discontinued. The third sum accounts for the customers whose first-choice is product $i-2$ and substitute to product i because neither their first nor second choice products are available. By the same logic, we can define $w_1(\tau_1)$ and $w_2(\tau_1, \tau_2)$ by taking the first and first two terms in the above summation respectively. This gives another method to denote our assortment optimization problem:

$$\max_{\tau \in \{0, \dots, T+1\}^n} w_1(\tau_1) + w_2(\tau_1, \tau_2) + \sum_{i=3}^n w_i(\tau_{i-2}, \tau_{i-1}, \tau_i). \quad (4.3)$$

Finally, for $i \geq 2$, let $V_i(\tau_{i-1}, \tau_i)$ be the maximum profit that we earn from products 1 through i , given the product discontinuation times τ_{i-1} and τ_i . In

other words, we want to maximize the profits from products 1 through i by choosing the product discontinuation times for products 1 through $i - 2$:

$$V_i(\tau_{i-1}, \tau_i) = \max_{\substack{\tau_j \in \{0, \dots, T+1\}, \\ j=1, \dots, i-2}} w_1(\tau_1) + w_2(\tau_1, \tau_2) + \sum_{j=3}^i w_j(\tau_{j-2}, \tau_{j-1}, \tau_j).$$

Solving Problem (4.3) is equivalent to solving $\max_{(\tau_{n-1}, \tau_n) \in \{0, \dots, T+1\}^2} V_n(\tau_{n-1}, \tau_n)$.

Our base case is $i = 2$. At $i = 2$, we have $V_2(\tau_1, \tau_2) = w_1(\tau_1) + w_2(\tau_1, \tau_2)$. For $i \geq 3$, we can rewrite $V_i(\tau_{i-1}, \tau_i)$ to obtain the following dynamic program:

$$\begin{aligned} V_i(\tau_{i-1}, \tau_i) &= \max_{\substack{\tau_j \in \{0, \dots, T+1\}, \\ j=1, \dots, i-2}} w_1(\tau_1) + w_2(\tau_1, \tau_2) + \sum_{j=3}^i w_j(\tau_{j-2}, \tau_{j-1}, \tau_j) \\ &= \max_{\tau_{i-2} \in \{0, \dots, T+1\}} w_i(\tau_{i-2}, \tau_{i-1}, \tau_i) + \\ &\quad \left(\max_{\substack{\tau_j \in \{0, \dots, T+1\}, \\ j=1, \dots, i-3}} w_1(\tau_1) + w_2(\tau_1, \tau_2) + \sum_{j=3}^{i-1} w_j(\tau_{j-2}, \tau_{j-1}, \tau_j) \right) \\ &= \max_{\tau_{i-2} \in \{0, \dots, T+1\}} w_i(\tau_{i-2}, \tau_{i-1}, \tau_i) + V_{i-1}(\tau_{i-2}, \tau_{i-1}). \end{aligned}$$

For each product, this dynamic program has $O(T^2)$ states and each state considers $O(T)$ decisions. Hence, solving Problem (4.3) takes $O(nT^3)$ operations via dynamic programming.

To extend the dynamic program for larger K , observe that we need to know the availability times of products $i - K + 1, \dots, i - 1, i$ in order to determine the profit earned by selling product i . We can define w_i similarly as before, counting over all the customers who purchase product i either as a first choice or through substituting. The customers who purchase product i are:

$$w_i(\tau_{i-K+1}, \dots, \tau_{i-1}, \tau_i) = \pi_i \cdot \left(\sum_{j=i-K+1}^i \sum_{t=1}^T \mathbb{1}[i, j \in \sigma_t, j-1 \notin \sigma_t, \max\{\tau_j, \dots, \tau_{i-1}\} < t \leq \tau_i] \right).$$

Then the dynamic program V_i would take $(\tau_{i-K+2}, \dots, \tau_i)$ as the state variable, with $O(T^{K-1})$ states per product. Each state considers at most $O(T)$ possible de-

cisions. When customers consider at most K products, we need $O(nT^K)$ operations to compute the optimal product discontinuation times via dynamic programming.

Theorem 8. *Suppose customers have preference lists which are intervals over a central ranking of the products, and customers consider at most K products. For any fixed K , we can compute the optimal assortment to offer and the optimal time to discontinue the products from the store in $O(nT^K)$ operations via dynamic programming. Hence, we can recover the optimal stocking quantity in the same runtime in the deterministic setting.*

4.4 Dynamic Substitution in the Stochastic Setting

We leverage our work in the deterministic setting to develop a heuristic for the stochastic setting. In the stochastic setting, the customers arrival sequence and their preference lists are unknown. Instead, this information is randomly drawn from some finite space Ω . We point out other difficulties in the stochastic setting and propose a heuristic based on our work in the deterministic setting.

4.4.1 Relationship Between the Deterministic and the Stochastic Settings

We identify three main difficulties of developing an efficient algorithm to compute the optimal assortment with stocking levels in the stochastic setting. The first difficulty is that the revenues and costs of products are relevant in the stochastic setting. Stocking a product with a high cost could be risky if it has

low demand, and a retailer may choose against stocking such a product even when it produces high profit. In the deterministic setting, this was not an issue and only the profits matter.

The second difficulty is that it is unclear how we can write the expected revenue function given the assortment and initial stocking quantities. The preference list of customer t and the assortment seen by the t -th customer upon her arrival are both random quantities. To the best of our knowledge, it is not known how the expected revenue can be computed efficiently for a given vector \mathbf{u} . Computing the expected revenue via dynamic programming would require the size of the state space to grow exponentially in the number of products. Previous literature have turned to simulation when the distribution of customers arrival sequences satisfy certain properties (Goyal et al. (2016), Aouad, Levi and Segev (2015b)).

Finally, our strategy of considering the product discontinuation times does not translate smoothly. Given a fixed time to discontinue product i from the store, the units of product i sold is still a random quantity. Alternatively, given a stocking level of product i , the stock-out time of the product i is also a random quantity. There is no mapping between the stocking quantity view of our problem versus the product discontinuation view.

Let Ω be the universe of all customers arrival sequences. Then each outcome $\omega \in \Omega$ is a customers arrival sequence, and denotes the information on the number of customers who arrive over the selling horizon, their preference lists, and the order in which the customers arrive. If Ω is finite and has probability distribution P , then we can represent our assortment optimization problem with the following integer program. Under customers arrival sequence ω , we let T_ω

denote the number of customers who arrive, and use binary vector $\mathbf{x}_{\omega,t} \in \{0, 1\}^n$ to denote the purchasing decision of the t -th customer in this outcome. We construct the following integer program, where $T^* = \max_{\omega \in \Omega} T_\omega + 1$:

$$\begin{aligned}
& \max \sum_{\omega \in \Omega} \left(\sum_{t=1}^{T_\omega} \sum_{i=1}^n r_i x_{\omega,t,i} \right) \cdot P(\omega) - \sum_{i=1}^n c_i u_i & (4.4) \\
& s.t. \sum_{i=1}^n x_{\omega,t,i} \leq 1 & \forall \omega \in \Omega; t = 1, \dots, T_\omega \\
& \sum_{t=1}^{T_\omega} x_{\omega,t,i} \leq u_i & \forall \omega \in \Omega; i = 1, \dots, n \\
& u_j - \sum_{t'=1}^{t-1} x_{\omega,t',j} \leq T^* \cdot (1 - x_{\omega,t,i}) & \forall \omega \in \Omega; t = 1, \dots, T_\omega \\
& & i, j \in \sigma_{\omega,t} : j >_{\omega,t} i \\
& x_{\omega,t,i} = 0 & \forall \omega \in \Omega; t = 1, \dots, T_\omega; i \notin \sigma_{\omega,t} \\
& \mathbf{x}_{\omega,t} \in \{0, 1\}^n & \forall \omega \in \Omega; t = 1, \dots, T_\omega \\
& \mathbf{u} \geq 0, \mathbf{u} \in \mathbb{Z}^n.
\end{aligned}$$

The reason that we need to use T^* instead of T_ω in the constraint governing product substitution is due to the central decision \mathbf{u} . If we had used T_ω in the constraint instead, then the maximum number of units that we can stock of any product would be artificially upper-bounded by $\min\{T_\omega : \omega \in \Omega\}$. Any T^* which upper-bounds T_ω would work in Problem (4.4). It is also worth noting that the same stocking costs apply over all $\omega \in \Omega$, but the revenue depends on the outcome. Furthermore, the vectors $\mathbf{x}_{\omega,t}$ are fixed when we are given \mathbf{u} .

Of course, Problem (4.4) is not efficient to solve directly. Instead, we introduce Problem (4.4) so that we can benchmark and measure the performance of our heuristic.

4.4.2 Heuristic: Best-of-the-Best

For customers arrival sequence $\omega \in \Omega$, let $\Pi(\mathbf{u}, \omega)$ denote the profit earned when the retailer offers \mathbf{u} . In other words, $\Pi(\mathbf{u}, \omega)$ is the objective value of Problem (4.1) under solution \mathbf{u} when the realized customers arrival sequence is ω . If we have a sample set \mathcal{S} of Ω , then $\Pi(\mathbf{u}, \mathcal{S})$ denotes the average profit over \mathcal{S} . That is:

$$\Pi(\mathbf{u}, \mathcal{S}) = \frac{1}{|\mathcal{S}|} \sum_{\omega \in \mathcal{S}} \Pi(\mathbf{u}, \omega).$$

Our heuristic works by sampling L realizations of Ω : $\mathcal{S} = \{\omega^1, \dots, \omega^L\}$, where L is a large number. For each of these realizations ω^ℓ , we compute an assortment with stocking level \mathbf{u}^ℓ under the deterministic setting. In order for the heuristic to be efficient, \mathbf{u}^ℓ does not necessarily have to be an optimal solution under realization ω^ℓ . When Ω consists of customers arrival sequences such that all customers are K -choosy, we can use Algorithm 3 in Subsection 4.3.3 to compute product discontinuation times τ^ℓ which achieves $\frac{2}{K} \cdot (1 - K)^{K-1}$ -fraction of the optimal profit. When Ω consists of customers arrival sequences such that all customers have preference lists which are intervals on a central ranking, then we can use the dynamic program in Subsection 4.3.4 to compute the optimal product discontinuation times τ^ℓ . We can apply Lemma 6 to recover \mathbf{u}^ℓ . We choose the \mathbf{u}^ℓ that generates the highest profit when averaged over \mathcal{S} . We call this solution \mathbf{u}^B because it is the best-of-the-best solutions:

$$\mathbf{u}^B = \arg \max_{1 \leq \ell \leq L} \Pi(\mathbf{u}^\ell, \mathcal{S}).$$

Intuitively, even though \mathbf{u}^ℓ was computed without considering the costs of products, computing its profit over all the realizations in \mathcal{S} would incur the

average cost of unsold units and identify risky products which have high cost and low demand.

4.4.3 Upper-bound on Expected Profit over Sample Space

Suppose our best-of-the-best heuristic returns a solution \mathbf{u}^B , and we want to test the quality of our solution when we are given a sample set $S \subset \Omega$. We want to compare the optimal average profit over S against the average profit earned with \mathbf{u}^B , and obtain some guarantee that we are not far from optimality. That is, we want to measure $\Pi(\mathbf{u}^B, S) / \max_{\mathbf{u} \in \mathbb{Z}^n} \Pi(\mathbf{u}, S)$.

We can upper-bound the value of $\max_{\mathbf{u} \in \mathbb{Z}^n} \Pi(\mathbf{u}, S)$ to obtain a lower-bound on the performance of our heuristic. To achieve our desired upper-bound, we can modify Problem (4.4) to maximize the average profit over S . Consider the following integer program:

$$\begin{aligned}
& \max \frac{1}{|S|} \sum_{\omega \in S} \left(\sum_{t=1}^{T_\omega} \sum_{i=1}^n r_i x_{\omega,t,i} \right) - \sum_{i=1}^n c_i u_i & (4.5) \\
& s.t. \sum_{i=1}^n x_{\omega,t,i} \leq 1 & \forall \omega \in S; t = 1, \dots, T_\omega \\
& \sum_{t=1}^{T_\omega} x_{\omega,t,i} \leq u_i & \forall \omega \in S; i = 1, \dots, n \\
& u_j - \sum_{t'=1}^{t-1} x_{\omega,t',j} \leq T^* \cdot (1 - x_{\omega,t,i}) & \forall \omega \in S; t = 1, \dots, T_\omega \\
& & i, j \in \sigma_{\omega,t} : j >_{\omega,t} i \\
& x_{\omega,t,i} = 0 & \forall \omega \in S; t = 1, \dots, T_\omega; i \notin \sigma_{\omega,t} \\
& \mathbf{x}_{\omega,t} \in \{0, 1\}^n & \forall \omega \in S; t = 1, \dots, T_\omega \\
& \mathbf{u} \geq 0, \mathbf{u} \in \mathbb{Z}^n.
\end{aligned}$$

There are two difference between Problems (4.4) and (4.5). First, the constraints are now limited to $\omega \in \mathcal{S}$, which is a subspace of Ω . Second, the objective function is now a sample average profit over \mathcal{S} , rather than the mean profit under Ω . Although solving Problem (4.5) would give us the value of $\max_{\mathbf{u} \in \mathbb{Z}^n} \Pi(\mathbf{u}, \mathcal{S})$, it is very difficult to solve because it is a large integer program with $O(nT^*|\mathcal{S}|)$ variables and $O(T^*|\mathcal{S}|)$ constraints. We solve its linear programming relaxation to provide an upper-bound on $\max_{\mathbf{u} \in \mathbb{Z}^n} \Pi(\mathbf{u}, \mathcal{S})$. Let $\Pi^{\text{LP}}(\mathcal{S})$ denote the optimal objective value of the linear programming relaxation of Problem (4.5). Then $\Pi^{\text{LP}}(\mathcal{S}) \geq \max_{\mathbf{u} \in \mathbb{Z}^n} \Pi(\mathbf{u}, \mathcal{S})$.

4.5 Numerical Study

There are two parts to our numerical experiments. First, we want to test the performance of our approximation algorithm for K -choosy customers in the deterministic setting. We focus on $K = 2$ for our numerical experiments, and refer to 2-choosy customers as choosy customers. We want to show that the solution from our approximation algorithm performs much better than its theoretical guarantee of $1/2$ of optimal profit. This is an important step because our heuristic uses Algorithm 3 to efficiently compute a solution \mathbf{u}^ℓ for $\omega^\ell \in \mathcal{S}$ when Ω consists of customers arrival sequences such that all customers are choosy. If the approximation algorithm performs poorly, then it would not make sense to use it as a subroutine in our heuristic.

Second, we want to test the performance of our heuristic introduced in Section 4.4 when customers are choosy or have preference lists that are intervals over the central ranking. For choosy customers, we use our 2-approximation al-

gorithm as our subroutine. For customers with preference lists that are intervals over the central ranking, we use our dynamic program as our subroutine.

In this section, we first describe how products are generated. Then, we describe how we construct the space of customers arrival sequences and the probability distribution over Ω .

4.5.1 Generating Products

We describe four ways in which we generate the products, from which the retailer selects his assortment.

Independent Costs and Profits: The cost of product i , c_i , is generated uniformly over $[0, 100]$. The profit of product i , π_i , is also generated uniformly over $[0, 100]$. The revenue of product i , r_i , is the cost plus profit: $r_i = \pi_i + c_i$.

High-Risk, High-Reward: The cost of product i , c_i , is generated uniformly over $[0, 100]$ and then squared, so that the variance in costs is high amongst the products. The profit of product i is $\pi_i = 0.5c_i$, so that its revenue is $r_i = 1.5c_i$. The fixed profit margin ensures that products which have a high cost to stock will also bring a higher profit when it is sold.

Similar Products: The cost of product i , c_i , is generated uniformly over $[0, 100]$ and then square-rooted, so that the variance in costs is low amongst the products. The profit of product i is $\pi_i = 0.5c_i$, so that its revenue is $r_i = 1.5c_i$. The fixed profit margin ensures that products are similar in terms of costs and profits.

Increasing Costs: The cost of product i , c_i , is generated uniformly over $[0, 100]$.

The profit of product i is $\pi_i = 0.5c_i$, so that its revenue is $r_i = 1.5c_i$. The products are sorted by increasing costs. This method of generating products is useful when there is a central ranking on the products which correlates prices with product quality. In this case, the central ranking would be $1 > 2 > \dots > n$, and each customer prefers the less expensive products in her preference list.

4.5.2 Generating Customer Arrivals Rates for Customers

The set Ω contains all possible customers arrival sequences over a finite selling horizon of T periods. Since we are focusing our tests on choosy customers and customers who consider intervals on a central ranking, we can describe customer types by (i, j) . For a choosy customer, i represents her first-choice product and j represents her second-choice product. For a customer who considers intervals on the central ranking, i represents her first-choice product and j represents her last-choice product, so that she ranks products as $i > i + 1 > \dots > j$. For each element $\omega \in \Omega$, the customers arrival sequence specifies whether a customer arrives at time $t = 1, \dots, T$, and the customer's type (i, j) conditional on her arrival.

To test our heuristic, we need to generate different probability distributions over Ω , compute \mathbf{u} for each distribution, and check that we perform well in expectation. We assume that customer arrivals are independent over time. Hence, we describe how we construct the probability distribution at each time period. Let λ denote the probability that a customer arrives at any time period. Conditional on a customer visiting the store, let $P_t(i, j)$ denote the the probability that a customer of type (i, j) arrives at time t . Finally, customer arrivals can

be homogenous or heterogeneous. If customers are homogenous over time, then $P_t(i, j) = P_1(i, j)$ for all (i, j) , and we only have to generate the probability distribution over the first selling period. If customers are heterogeneous over time, then we generate different probability distributions $P_t(\cdot, \cdot)$ for each period $t = 1, \dots, T$. We describe three ways in which we generate the probability distribution $P_t(\cdot, \cdot)$.

Choosy Customers

If a customer's type is (i, j) , then her first choice product is i and her second choice is j . For a fixed time period t , we first generate weights $w_{i,j}$ for each pair (i, j) such that $i, j \in N$. If $i = j$, then the customer is only interested in one product. The distribution $P_t(\cdot, \cdot)$ is defined such that $P_t(i, j) = w_{i,j} / \sum_{(i', j')} w_{i', j'}$.

Fully Non-Parametric: We allow customers to be interested in any two products, and allow the two products to be in any order. For each (i, j) such that $i, j \in N$, the weight $w_{i,j}$ is generated uniformly over $[0, 1]$.

Universal Substitute: Every customer who does not obtain her favourite product substitutes to the same product. The universal substitute product, j^* , can be seen as a generic product whereas the first-choice product is a specialized product. We first choose the substitute product j^* uniformly at random amongst the products available to the retailer. For each (i, j) such that $j = j^*$, the weight $w_{i,j}$ is generated uniformly over $[0, 1]$. If $j \neq j^*$, then $w_{i,j}$ is set to 0.

Fixed Ordering: We assume there is a central ranking on the products, so that $1 > 2 > \dots > n$, but customers can consider any two products. For each (i, j) such that $i < j$, the weight $w_{i,j}$ is generated uniformly over $[0, 1]$. Otherwise, $w_{i,j}$

is set to 0.

Customers with Interval Preference Lists

When we consider customers with preference lists which are intervals of a central ranking, we focus on customers who rank at most three products. The central ranking is $1 > 2 > \dots > n$. If a customer's type is (i, j) , then her preference list is $i > i + 1 > \dots > j$. For a fixed time period t , we first generate weights $w_{i,j}$ for each pair (i, j) such that $i, j \in N$, $j - i + 1 \leq 3$, and $i \leq j$. If $i = j$, then the customer is interested in only one product. The distribution $P_t(\cdot, \cdot)$ is defined such that $P_t(i, j) = w_{i,j} / \sum_{(i', j')} w_{i', j'}$.

General: Customers are interested in one to three products. For i, j such that $j - i + 1 \leq 3$ and $i \leq j$, the weight $w_{i,j}$ is generated uniformly over $[0, 1]$. This represents a customer with preference list $i > \dots > j$.

Exactly-3-Products: Customers are interested in exactly three products, $i > i + 1 > i + 2$, for $i \leq n - 2$. For $i \leq n - 2$, the weight $w_{i,i+2}$ is generated uniformly over $[0, 1]$. All other weights $w_{i,j}$ such that $j - i \neq 2$ is set to 0.

Not-Too-Picky: When customers are not too picky, customers are more likely to be interested in three products than only one product. To create a distribution of customers, the weight $w_{i,j}$ is generated uniformly over $[0, j - i + 1]$ when $j - i + 1 \leq 3$ and $i \leq j$. As a result, the mean weight of a customer who ranks three products is three times larger than the mean weight of a customer who ranks one product.

4.5.3 Structure of the Numerical Experiments

We define a setup to be the combination of one method for generating products in Subsection 4.5.1, one method of generating the probability distribution of customers $P_t(\cdot, \cdot)$ in Subsection 4.5.2, and the decision of whether customers are homogenous or heterogeneous over time. In accordance with the setup, an instance of our assortment optimization problem can be created with n products and a probability distribution over customers arrival sequences for a selling horizon of T periods. We set $n = 10$, $T = 100$, and the probability of no customers arriving at any time period to be $\lambda = 0.1$.

For each possible setup, we test our heuristic over 20 instances of the assortment optimization problem with dynamic substitution under the stochastic setting. For each instance of the assortment optimization problem, we generate a sample set of $L = 100$ realizations of customers arrival sequences using the probability distribution created above. We denote this sample set as $\mathcal{S}^{\text{train}}$. This sample set, $\mathcal{S}^{\text{train}}$, is used for two purposes: testing the approximation algorithm for choosy customers in the deterministic setting, and computing \mathbf{u}^B based on the heuristic for the stochastic setting. We describe these tests and our performance measures next.

4.5.4 Performance of the 2-Approximation Algorithm for Choosy Customers

The first test measures the performance of our 2-approximation algorithm for choosy customers in the deterministic setting. Unless the algorithm works sig-

nificantly better in practice than its theoretical guarantee, it does not make a lot of sense to use it as a subroutine.

For $\ell = 1, \dots, L$, we compute an assortment with stocking levels \mathbf{u}^ℓ when the retailer observes realization ω^ℓ by solving the linear programming relaxation of Problem (4.2) and rounding the solution according to Algorithm 3. We also solved the related integer program, Problem (4.2), using Gurobi for solution $\mathbf{u}^{\ell*}$. We measure the fraction of optimal profit earned as:

$$\Pi(\mathbf{u}^\ell, \omega^\ell) / \Pi(\mathbf{u}^{\ell*}, \omega^\ell).$$

Surprisingly, solving the linear programming relaxation of Problem (4.2) for choosy customers frequently resulted with almost integral solution. There were very few products that required rounding and the solution \mathbf{u}^ℓ found by our approximation algorithm came very close to optimality. Over the 20 instances with 100 realizations per instance, the measure above never fell below 98% of optimality. Hence, our approximation algorithm performs much better in practice than its theoretical guarantee.

4.5.5 Performance of the Heuristic for Stochastic Setting

The second test compares the performance of our heuristic against a theoretical upper-bound on the average profit over a sample set of Ω . For each probability distribution over Ω , we create two sample sets of equal sizes, $\mathcal{S}^{\text{train}}$ and $\mathcal{S}^{\text{test}}$. The set $\mathcal{S}^{\text{train}}$ is used to run our heuristic and compute an assortment and stocking levels \mathbf{u}^B . The set $\mathcal{S}^{\text{test}}$ is used to test the performance of \mathbf{u}^B .

We apply our best-of-the-best heuristic on the sample set $\mathcal{S}^{\text{train}}$ to obtain an assortment and stocking level \mathbf{u}^B . Note that when we run our heuristic, the

underlying subroutine is an approximation algorithm when we have choosy customers but an exact algorithm when we consider customers with preference lists that are intervals on a central ranking. However, based on our previous test, our approximation algorithm is near-optimal and never fell below 98% of the optimal objective value in the test from Subsection 4.5.4.

We test the performance of \mathbf{u}^B on $\mathcal{S}^{\text{test}}$. We compute $\Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})$, which is the average profit over $\mathcal{S}^{\text{test}}$ when we use the solution \mathbf{u}^B from our heuristic. To upper-bound the optimal average profit over $\mathcal{S}^{\text{test}}$, we solve the linear programming relaxation of Problem (4.5) with $\mathcal{S} = \mathcal{S}^{\text{test}}$. Let $\Pi^{\text{LP}}(\mathcal{S}^{\text{test}})$ be the optimal objective value of the linear programming relaxation. Clearly, $\Pi^{\text{LP}}(\mathcal{S}^{\text{test}}) \geq \Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})$, and we want to measure the ratio $\Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})/\Pi^{\text{LP}}(\mathcal{S}^{\text{test}})$.

Finally, we also want to test our heuristic against a naïve rounding heuristic based on the initial sample $\mathcal{S}^{\text{train}}$. A naïve approach is to solve the linear programming relaxation of Problem (4.5) with $\mathcal{S} = \mathcal{S}^{\text{train}}$ to obtain a solution $\bar{\mathbf{u}}$. We then round each \bar{u}_i probabilistically, so that we round up with probability $\bar{u}_i - \lfloor \bar{u}_i \rfloor$. This is a natural comparison as it is the simplest integral solution which we can obtain from knowing $\mathcal{S}^{\text{train}}$. Since our heuristic computes 100 solutions, \mathbf{u}^ℓ for $\ell = 1, \dots, 100$, and chooses the solution which achieves the highest average profit over $\mathcal{S}^{\text{train}}$, we also want to let the naïve rounding heuristic choose amongst 100 solutions. As a result, we round $\bar{\mathbf{u}}$ probabilistically 100 times and choose the integral solution which achieves the highest average profit over $\mathcal{S}^{\text{train}}$. We call the resulting solution \mathbf{u}^R and we measure the ratio $\Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})/\Pi(\mathbf{u}^R, \mathcal{S}^{\text{test}})$.

Note that $\Pi^{\text{LP}}(\mathcal{S}^{\text{test}})$ is a loose upper-bound on the optimal average profit over $\mathcal{S}^{\text{test}}$, so the value of $\Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})$ can never achieve the value of $\Pi^{\text{LP}}(\mathcal{S}^{\text{test}})$. On

the other hand, $\Pi(\mathbf{u}^R, \mathcal{S}^{\text{test}})$ is neither an upper- nor lower-bound on $\Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})$. Rather, it tests the effectiveness of our heuristic against a naïve heuristic that does not require the ability to solve the assortment optimization problem with dynamic substitution, even under the deterministic setting.

Results of heuristic: Choosy Customers

We look at the performance of our heuristic when all the customers are choosy customers. Table 4.1 summarizes the average performance of \mathbf{u}^B against our linear programming relaxation of Problem (4.5), over 20 instances of our problem. That is, we generated 20 different probability distributions over Ω and computed $\Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})/\Pi^{\text{LP}}(\mathcal{S}^{\text{test}})$. Table 4.2 summarizes the performance of \mathbf{u}^B against the naïve solution \mathbf{u}^R by giving the ratio $\Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})/\Pi(\mathbf{u}^R, \mathcal{S}^{\text{test}})$.

Given the simplicity of the best-of-the-best heuristic, we believe that our heuristic performs reasonably well. We discuss the results in three parts: customers with universal substitutes, customers who follow a fixed ordering on the products with increasing costs, and all other setups according to Tables 4.1 and 4.2.

When all customers have an universal substitute, then our heuristic performs very well and achieves more than 95% of the upper-bound value $\Pi^{\text{LP}}(\mathcal{S}^{\text{test}})$ on average. This is true regardless of how products are generated. However, the naïve approach of rounding the solution from a linear programming relaxation of Problem (4.5) also performs extremely well and our heuristic could not outperform the naïve approach. One observation is that the deterministic setting can be solved by inspection when customers have an universal substitute. The

	Products \ Customers	Fully Non-Parametric	Universal Substitute	Fixed Ordering
Homogenous	Independent c_i and π_i	.83	.98	.85
	High-Risk, High-Reward	.77	.96	.79
	Similar Products	.75	.94	.77
	Increasing Costs	.75	.96	.71
Heterogeneous	Independent c_i and π_i	.83	.98	.84
	High-Risk, High-Reward	.77	.96	.79
	Similar Products	.75	.94	.77
	Increasing Costs	.76	.95	.71

Table 4.1: Choosy Customer - Heuristic vs LP upper-bound $\Pi(\mathbf{u}^\beta, \mathcal{S}^{\text{test}}) / \Pi^{\text{LP}}(\mathcal{S}^{\text{test}})$

	Products \ Customers	Fully Non-Parametric	Universal Substitute	Fixed Ordering
Homogenous	Independent c_i and π_i	1.04	.99	1.07
	High-Risk, High-Reward	1.16	.96	1.19
	Similar Products	1.03	.94	1.12
	Increasing Costs	1.09	.96	1.49
Heterogeneous	Independent c_i and π_i	1.04	.99	1.08
	High-Risk, High-Reward	1.17	.96	1.22
	Similar Products	1.02	.94	1.13
	Increasing Costs	1.09	.95	1.53

Table 4.2: Choosy Customer - Heuristic vs naïve rounding $\Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})/\Pi(\mathbf{u}^R, \mathcal{S}^{\text{test}})$

retailer should not offer any products with profit less than the universal substitute j^* , and the optimal stocking level ensures that customers buy their first choice product i if $\pi_i \geq \pi_{j^*}$ and substitute to j^* otherwise. The simplicity of the substitution pattern may explain why both heuristics perform extremely well.

When products increase in cost and profit, and customers obey a fixed ordering on their preferences so that $1 > \dots > n$, our heuristic performs poorly when we compare our results to the linear programming relaxation upper-bound. In fact, the average of $\Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})/\Pi^{\text{LP}}(\mathcal{S}^{\text{test}})$ is only about 70%. However, we outperform the naïve heuristic significantly, and $\Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})$ is about 1.5 times the value of $\Pi(\mathbf{u}^R, \mathcal{S}^{\text{test}})$. Rounding the linear programming relaxation's optimal solution resulted in a heavy loss in the objective value of Problem (4.5).

We focus on the disparity between the optimal objective value to the linear programming relaxation of Problem (4.5) and the average profit earned by the rounded solution \mathbf{u}^R to understand why our heuristic performs poorly against the linear programming relaxation upper-bound but very well against the naïve heuristic. One reason for this disparity is that the linear programming relaxation of Problem (4.5) overestimates the retailer's ability to sell product 10, which has the highest cost, profit, and revenue. To see why this is true, observe that customers always have a preference for a less-expensive option. As such, product 10 is never sold unless the customer's first choice product has sold out. However, the linear programming relaxation allows variables $x_{\omega,t,10}$ to be positive even when a preferred product is still in stock. The linear programming relaxation overstates the retailer's ability to sell product 10, and this overstates the revenue earned by product 10. When we consider the actual profit earned by \mathbf{u}^R , we see instead that the unsold units of product 10 results in a high stocking cost

which is not offset by the high revenue that it promises to earn. As evidence, we looked at \bar{u}_{10} , where $\bar{\mathbf{u}}$ is the optimal solution to the linear programming relaxation of Problem (4.5) with $\mathcal{S} = \mathcal{S}^{\text{train}}$. Over the 20 realizations of $\omega \in \Omega$, we saw that \bar{u}_{10} is about 27% greater than the average u_{10}^ω with homogenous customers and 32% greater with heterogeneous customers. This supports our hypothesis that the linear programming relaxation of Problem (4.5) under $\mathcal{S}^{\text{train}}$ and its rounded solution \mathbf{u}^R overstates the retailer's ability to sell the product with both the highest cost and highest profit.

Under the remaining setups in Tables 4.1 and 4.2, we achieve 75-85% of $\Pi^{\text{LP}}(\mathcal{S}^{\text{test}})$ on average, and performed better than the naïve rounding heuristic by 2-22%. This is surprisingly good considering that our algorithm is an intuitive method for making the most use of an algorithm meant for the deterministic setting.

Results of heuristic: Customers with Interval Preference Lists

We look at the performance of our heuristic when all the customers have preference lists which are intervals on a central ranking. Table 4.3 summarizes the average performance of \mathbf{u}^B against our linear programming relaxation of Problem (4.5), over 20 instances of our problem. That is, we generated 20 different probability distributions over Ω and computed $\Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})/\Pi^{\text{LP}}(\mathcal{S}^{\text{test}})$. Table 4.4 summarizes the performance of our \mathbf{u}^B against the naïve solution \mathbf{u}^R by giving the ratio $\Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})/\Pi(\mathbf{u}^R, \mathcal{S}^{\text{test}})$.

When we look at customers with preference lists that are intervals over the central ranking, there is no equivalent concept of customers having an universal

substitute. If we divide our results in two groups, we observe similar patterns with the results for choosy customers. We focus on the case where products have increasing costs, and group the remaining setups listed in Tables 4.3 and 4.4.

When products have increasing costs, the results are similar to choosy customers following a fixed ordering. The reason is that the products with higher cost and profits are always ranked lower by customers. We achieve only 72-85% of the guarantee on optimal expected profit when we consider the average of $\Pi(\mathbf{u}^B, \mathcal{S}^{\text{test}})/\Pi^{\text{LP}}(\mathcal{S}^{\text{test}})$, but we outperform the naïve rounding heuristic by 9-169%. We observed that the linear programming relaxation overstates the retailer's ability to sell a last-choice product to customers, which has a high profit when sold but incurs a high cost when stocked without being sold. Again, we verify this reasoning by focusing on product 10, which is always a last-choice product. We compared \bar{u}_{10} from solving the linear programming relaxation of Problem (4.5) to the deterministic solution u_{10}^ω over the 20 realizations of $\omega \in \Omega$, and found that \bar{u}_{10} is about 40% greater than the average value of u_{10}^ω . This is especially noticeable when customers always consider purchasing two other products before substituting to their expensive last-choice product. In summary, when customers rank products from least to most expensive and profits increase with costs, then solving the linear programming relaxation neither provides good bounds nor good heuristics. This offers support for using our heuristic rather than a more naïve approach.

For all the other problem setups listed in Tables 4.3 and 4.4, the performance of our heuristic are similar and we achieve about 78-95% of the upper-bound on optimal profit. We also perform well against the naïve heuristic. There is a

	Customers Products		General	Exactly-3	Not-Too-Picky
Homogenous	Independent c_i and π_i		.88	.95	.90
	High-Risk, High-Reward		.86	.93	.88
	Similar Products		.81	.91	.83
	Increasing Costs		.75	.85	.76
Heterogeneous	Independent c_i and π_i		.88	.95	.90
	High-Risk, High-Reward		.84	.92	.87
	Similar Products		.78	.90	.82
	Increasing Costs		.72	.83	.73

Table 4.3: Customers with interval preference lists - Heuristic vs LP upper-bound $\Pi(\mathbf{u}^B, \mathbf{S}^{\text{test}})/\Pi^{\text{LP}}(\mathbf{S}^{\text{test}})$

	Products \ Customers	General	Exactly-3	Not-Too-Picky
Homogenous	Independent c_i and π_i	1.06	1.17	1.07
	High-Risk, High-Reward	1.17	1.17	1.20
	Similar Products	1.06	1.14	1.10
	Increasing Costs	1.17	1.98	1.42
Heterogeneous	Independent c_i and π_i	1.07	1.07	1.07
	High-Risk, High-Reward	1.20	1.16	1.20
	Similar Products	1.07	1.12	1.12
	Increasing Costs	1.09	2.69	1.28

Table 4.4: Customers with interval preference lists - Heuristic vs naïve rounding $\Pi(\mathbf{u}^B, \mathbf{S}^{\text{test}})/\Pi(\mathbf{u}^R, \mathbf{S}^{\text{test}})$

slight dip in performance when products are similar in terms of cost and profit.

4.6 Conclusion

In this chapter, we study the assortment optimization problem with dynamic substitution and stocking costs. We show that the problem is NP-hard even when the customers arrival sequence is deterministic and known to the retailer. We focus on studying two classes of customers: K -choosy customers and customers who have preference lists in the form of intervals over a central ranking. Using our algorithms from the deterministic setting, we develop a heuristic which is intuitive and based on choosing the best deterministic solution when we take a sample set of customers arrival sequences.

Even though previous literature focused on the shelf-space constraint rather than stocking costs, we faced a similar challenge in our paper. Specifically, given some assortment and stocking levels \mathbf{u} , we do not know how to efficiently compute the expected number of units sold for each product. We used simulations in our numerical experiments to compute the expected revenue portion of our profit. An important future research direction when considering revenue management problems with dynamic substitution is to find ways to compute the expected revenue efficiently and exactly.

Another future direction is to consider provably-good algorithms for our problem under the stochastic setting. Our heuristic performed reasonably well, but we could not provide theoretical bounds on its performance. It is useful to study dynamic substitution with stocking costs under other choice models as well, such as MNL. Under the right choice model, it may be possible to leverage

the strategy of using product discontinuation times. This strategy helped immensely in the deterministic setting but we could not apply it in the stochastic setting.

CHAPTER 5

CONCLUDING REMARKS

In this thesis, we consider three assortment optimization problems. In Chapter 2, we consider an assortment optimization problem in an omnichannel retail environment. Some customers observe the products offered in the physical store and update their preferences for the products offered online, based on the features they observed in-store. In Chapter 3, we consider synergy effects in assortment optimization. This problem recognizes a marketing phenomenon where certain products look more attractive when offered alongside alternatives of lower quality. In Chapter 4, we consider assortment optimization with dynamic substitution and stocking costs. This problem recognizes that many retailers do not have the ability to adjust the assortment for each incoming customer, and that a customer makes a purchasing decision based on the assortment of products which have not stocked out.

Each of the problems studied in this thesis is meant to better model the current retail environment. We fill in some of the gaps between the models studied in the literature and real-world observations so that industry recognizes the value and applicability of assortment optimization techniques. As the retail environment continues to change, retailers will find more creative ways to interact with customers and incentivize purchasing. We expect many new and interesting problems to continue to emerge in this research space.

APPENDIX A

APPENDIX: OMNICHANNEL ASSORTMENT OPTIMIZATION UNDER THE MULTINOMIAL LOGIT MODEL WITH A FEATURES TREE

A.1 Proofs Omitted from Chapter 2

Proof of Claim 1: We only prove the first case because the other two cases are similar. Let \mathbf{x}^* be the optimal solution in the showroom setting, such that

$$\gamma^* = \frac{\sum_{i=1}^n \pi_i w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k^*}}{w_0 + \sum_{i=1}^n w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k^*}} > \gamma.$$

We can multiply both sides by the denominator as $w_0 > 0$ and rearrange terms to obtain:

$$\sum_{i=1}^n (\pi_i - \gamma) w_i \cdot \prod_{k \in A(i)} \delta_k^{x_k^*} > w_0 \gamma.$$

The left side of the above inequality is upper-bounded by $f(\gamma)$. □

Proof of Lemma 1: Since \mathbf{x} is integral and $x_{p(k)} \geq x_k$ for all $k \in A(i) \setminus \{\text{root}\}$, either $x_i = 1$, $x_{\text{root}} = 0$, or there exists some \bar{k} such that $x_{\bar{k}} = 0$ and $x_{p(\bar{k})} = 1$ for $\bar{k} \in A(i) \setminus \{\text{root}\}$. The result clearly holds for the first two cases, so we proceed to the third case. Then $x_k = 1$ for all $k \in A(p(\bar{k}))$ and $x_k = 0$ for $k \in A(i) \setminus A(p(\bar{k}))$, and the right side simplifies to:

$$\begin{aligned} w_i + w_i \cdot \sum_{k \in A(i)} (\Delta_k - \Delta_{p(k)}) \cdot x_k &= w_i + w_i \cdot \sum_{k \in A(p(\bar{k}))} (\Delta_k - \Delta_{p(k)}) \\ &= w_i + w_i \cdot \Delta_{p(\bar{k})} - w_i \\ &= w_i \prod_{k \in A(p(\bar{k}))} \delta_k \\ &= w_i \prod_{k \in A(i)} \delta_k^{x_k}. \end{aligned}$$

Therefore, we can rewrite $P_i^{ON}(\mathbf{x})$ and the expected online revenue as a fraction of linear functions.

$$P_i^{ON}(\mathbf{x}) = \frac{w_i + w_i \cdot \sum_{k \in A(i)} (\Delta_k - \Delta_{p(k)}) x_k}{w_0 + \sum_{j=1}^n \left(w_j + w_j \cdot \sum_{k \in A(j)} (\Delta_k - \Delta_{p(k)}) x_k \right)},$$

$$\begin{aligned} \sum_{i \in N} \pi_i P_i^{ON}(\mathbf{x}) &= \frac{\sum_{i=1}^n \pi_i w_i + \sum_{i=1}^n \pi_i w_i \cdot \sum_{k \in A(i)} (\Delta_k - \Delta_{p(k)}) x_k}{w_0 + \sum_{i=1}^n w_i + \sum_{i=1}^n w_i \cdot \sum_{k \in A(i)} (\Delta_k - \Delta_{p(k)}) x_k} \\ &= \frac{\sum_{i=1}^n \pi_i w_i + \sum_{k=1}^{2n-1} \left(\sum_{i \in L(k)} \pi_i w_i \right) \cdot (\Delta_k - \Delta_{p(k)}) \cdot x_k}{w_0 + \sum_{i=1}^n w_i + \sum_{k=1}^{2n-1} \left(\sum_{i \in L(k)} w_i \right) \cdot (\Delta_k - \Delta_{p(k)}) \cdot x_k}. \end{aligned}$$

The last equality switches the summation from products to vertices in the features tree T . □

Proof of Proposition 1: The OCA problem under the features tree model is in NP: given characteristic vector \mathbf{x} , we can easily determine if “ $\Pi(\mathbf{x}) \geq K$ ” for any K . By contradiction, suppose that for any instance of our problem and any K , we can determine if the optimal expected revenue is at least K . We will show NP-hardness via a reduction from the partition problem.

In the partition problem, we are given integers $\{c_1, \dots, c_n\}$ such that $\sum_{i=1}^n c_i = 2C$ for some integer C . The problem requires us to determine if there exists a subset $S \subseteq N$ such that $\sum_{i \in S} c_i = C$. Our proof constructs an instance of the OCA problem such that an optimal assortment will give us a solution to the partition problem if it exists.

The portion of online customers is $q = \frac{2C^2+C}{5C^2+4C+1} + \frac{1}{2}$, and the portion of offline customers is $1 - q = \frac{3C^2+3C+1}{5C^2+4C+1} - \frac{1}{2}$. It is easy to see that $q \in (0, 1)$ for any $C \geq 1$.

Construct n products, each with revenue $\pi_i = 1$. An online customer associates an initial preference weight $w_i = 2c_i$ to product i . It will be easier to

consider a general tree rather than a binary tree for the purpose of this proof. Our tree has $n + 1$ vertices, consisting of n leaves and a root. Each leaf has a multiplier $\delta_i = 1/2$, and the root has a multiplier $\delta_{\text{root}} = 1$. Since $\delta_{\text{root}} = 1$, it never affects online preference weights and we can ignore x_{root} . Our feasible region reduces to $\mathcal{X} = \{0, 1\}^n$. Given $\mathbf{x} \in \{0, 1\}^n$, the resulting preference weights of online customers for product i is $v_i(\mathbf{x}) = 2c_i \left(\frac{1}{2}\right)^{x_i} = c_i(2 - x_i)$. The second equality can be obtained by applying Lemma 1. An offline customer has preference weight $\hat{v}_i = c_i$ for product i . Both customer types have a preference weight of 1 for the no-purchase option.

We claim that a partition of $\{c_i \mid i \in N\}$ exists if and only if the optimal expected revenue is greater or equal to $K = (5C^2 + 2C)/(5C^2 + 4C + 1)$. The expected revenue of $\mathbf{x} \in \{0, 1\}^n$ is:

$$\begin{aligned}\Pi(\mathbf{x}) &= q \cdot \frac{\sum_{i=1}^n \pi_i v_i(x)}{1 + \sum_{i=1}^n v_i(x)} + (1 - q) \cdot \frac{\sum_{i=1}^n \pi_i \hat{v}_i x_i}{1 + \sum_{i=1}^n \hat{v}_i x_i} \\ &= q \cdot \frac{\sum_{i=1}^n c_i(2 - x_i)}{1 + \sum_{i=1}^n c_i(2 - x_i)} + (1 - q) \cdot \frac{\sum_{i=1}^n c_i x_i}{1 + \sum_{i=1}^n c_i x_i} \\ &= q \cdot \frac{4C - \sum_{i=1}^n c_i x_i}{1 + 4C - \sum_{i=1}^n c_i x_i} + (1 - q) \cdot \frac{\sum_{i=1}^n c_i x_i}{1 + \sum_{i=1}^n c_i x_i}.\end{aligned}$$

The last line is true because $\sum_{i=1}^n c_i = 2C$. Let $y = 2C - \sum_{i=1}^n c_i x_i$ and rewrite the expected revenue as a function $f(y)$, with q expanded out:

$$f(y) = \left(\frac{2C^2 + C}{5C^2 + 4C + 1} + \frac{1}{2} \right) \cdot \frac{2C + y}{1 + 2C + y} + \left(\frac{3C^2 + 3C + 1}{5C^2 + 4C + 1} - \frac{1}{2} \right) \cdot \frac{2C - y}{1 + 2C - y}.$$

The first and second derivatives are:

$$f'(y) = \left(\frac{2C^2 + C}{5C^2 + 4C + 1} + \frac{1}{2} \right) \cdot \frac{1}{(1 + 2C + y)^2} - \left(\frac{3C^2 + 3C + 1}{5C^2 + 4C + 1} - \frac{1}{2} \right) \cdot \frac{1}{(1 + 2C - y)^2}, \text{ and}$$

$$f''(y) = - \left(\frac{2C^2 + C}{5C^2 + 4C + 1} + \frac{1}{2} \right) \cdot \frac{2}{(1 + 2C + y)^3} - \left(\frac{3C^2 + 3C + 1}{5C^2 + 4C + 1} - \frac{1}{2} \right) \cdot \frac{2}{(1 + 2C - y)^3}.$$

Since $f''(y) < 0$ for all $y \in [0, 2C]$, any solution \bar{y} such that $f'(\bar{y}) = 0$ must be a unique maximum over this interval. We can verify that $f'(C) = 0$ and that:

$$\begin{aligned} f(C) &= \left(\frac{2C^2 + C}{5C^2 + 4C + 1} + \frac{1}{2} \right) \cdot \frac{3C}{1 + 3C} + \left(\frac{3C^2 + 3C + 1}{5C^2 + 4C + 1} - \frac{1}{2} \right) \cdot \frac{C}{1 + C} \\ &= \frac{5C^2 + 2C}{5C^2 + 4C + 1} = K. \end{aligned}$$

The optimal expected revenue is upper-bounded by $f(C) = K$ and this is a unique maximum on the interval $[0, 2C]$. Rearranging $C = y = 2C - \sum_{i=1}^n c_i x_i$ gives us $C = \sum_{i=1}^n c_i x_i = \sum_{i \in S} c_i$. Hence the optimal expected revenue is greater or equal to K if and only if there exists some partition S such that $\sum_{i \in S} c_i = C$. This shows the OCA problem under the features tree model is NP-hard in the general setting. \square

A.2 Faster Algorithm via Longest Path in a Directed Acyclic Graph

One of the biggest contribution to the high runtime of our dynamic program in the general setting is the number of decisions that are considered. In this section, we show a constrained longest path problem on a directed acyclic graph (DAG) that is equivalent to solving Problems (2.1) and (2.3). This structure reduces the number of decisions per state and significantly reduces the runtime of our FPTAS by a factor of $O(n^2/\epsilon^2)$. We explain the intuition at the end of the section.

In the constrained longest path problem, we are given a graph G with a source s and a sink t . The arcs of G have costs \mathbf{c} , and L sets of weights $\mathbf{d}^1, \dots, \mathbf{d}^L$ with corresponding budget D^l . Let $c(P)$ and $d^l(P)$ denote the length of P under

costs \mathbf{c} and weights \mathbf{d}^l respectively: $c(P) = \sum_{(u,v) \in P} c_{u,v}$ and $d^l(P) = \sum_{(u,v) \in P} d_{u,v}^l$. Then the problem is to find a s - t path P that maximizes $c(P)$ subject to $d^l(P) \leq D_l$ for all $l = 1, \dots, L$.

We proceed by showing the longest path construction for Problem (2.1). We present a DAG G that has a bijection between every s - t path and solution $\mathbf{x} \in \mathcal{X}$. As a result, we can efficiently compute $f(\gamma)$ by finding the longest path in G via the standard dynamic program. The extension to the general setting simply requires putting two set of weights on the arcs of G to form a constrained longest path problem. Even though the constrained longest path problem is NP-hard in general, our instance can be solved efficiently because we use the rounded parameters from Problem (2.3), which are integers of size $O(n/\epsilon)$.

A.2.1 Showroom Setting

To solve Problem (2.1), we construct a directed graph $G = (V, E)$ with arc costs \mathbf{c}^γ based on our features tree T and parametrized for γ . For every vertex k in T , we will have two vertices in G : k_s and k_t . We also add a source s and a sink t into G . Hence, $V(G) = \{s, t\} \cup \{k_s, k_t \mid k \in V(T)\}$.

Let G_k be the subgraph of G induced by the set of vertices k'_s, k'_t where k' is a feature in the subtree rooted at k . The subgraphs are built recursively from the leaves of T up to the root. We start from $k \in N$ and create arc (k_s, k_t) with cost $c_{k_s, k_t}^\gamma = (\pi_k - \gamma)w_k\Delta_k$. So G_k has two vertices and one arc (see Figure A.1).

To build G_k where $k \notin N$, recall that k has two children: $\ell = \ell(k)$ and $r = r(k)$. Both G_ℓ and G_r have already been built, and we add five arcs to complete G_k (see

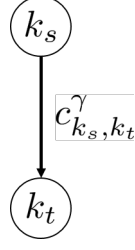


Figure A.1: G_k when $k \in N$, with arc cost $c_{k_s, k_t}^\gamma = (\pi_k - \gamma)w_k\Delta_k$.

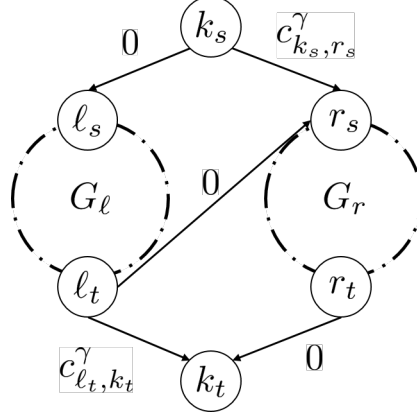


Figure A.2: G_k when $k \notin N$, with arc costs $c_{k_s, r_s}^\gamma = \Delta_k \cdot \left(\sum_{i \in L(\ell)} (\pi_i - \gamma)w_i \right)$ and $c_{\ell_t, k_t}^\gamma = \Delta_k \cdot \left(\sum_{i \in L(r)} (\pi_i - \gamma)w_i \right)$.

Figure A.2). In particular, arcs (k_s, ℓ_s) and (k_s, r_s) go from the source of G_k to the sources of G_ℓ and G_r respectively. Arcs (ℓ_t, k_t) and (r_t, k_t) go from the sinks of G_ℓ and G_r to the sink of G_k . We also join the sink of G_ℓ to the source of G_r with arc (ℓ_t, r_s) . Arcs (k_s, ℓ_s) , (ℓ_t, r_s) , and (r_t, k_t) have 0 costs. The other two arcs have costs:

$$c_{k_s, r_s}^\gamma = \Delta_k \cdot \left(\sum_{i \in L(\ell)} (\pi_i - \gamma)w_i \right), \quad \text{and} \quad c_{\ell_t, k_t}^\gamma = \Delta_k \cdot \left(\sum_{i \in L(r)} (\pi_i - \gamma)w_i \right).$$

Intuitively, a unit of flow entering G_k corresponds to feature k being demonstrated in-store. This unit of flow can do one of three things. First, it can pass through G_ℓ and then G_r , which corresponds to both children features being demonstrated in-store. Second, it can pass through G_ℓ but not G_r . In this case, the cost of c_{ℓ_t, k_t}^γ is incurred, which accounts for the fixed contribution by the right

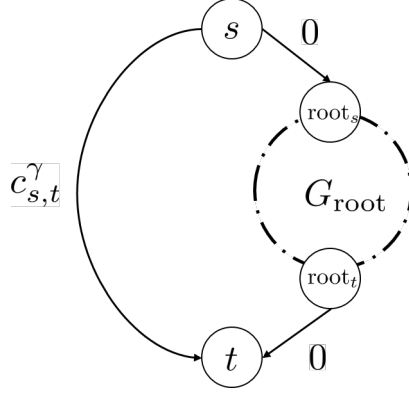


Figure A.3: DAG G , with arc cost $c_{s,t}^\gamma = \sum_{i=1}^n (\pi_i - \gamma) w_i$.

subtree when the left feature is demonstrated but not the right feature. Third, it can pass through G_r but not G_ℓ , in which case the cost of c_{k_s, r_s}^γ is incurred, and accounts for the fixed contribution by the left subtree. It is not possible for the unit of flow to neither enter G_ℓ nor G_r , which corresponds to demonstrating at least one child feature.

Finally, we connect G_{root} to the source s and sink t by adding arcs (s, root_s) , (root_t, t) , and (s, t) (see Figure A.3). The first two arcs have 0 costs and the third arc has $c_{s,t}^\gamma = \sum_{i=1}^n (\pi_i - \gamma) w_i$. Our graph G has $|V(G)| = 2 \cdot (2n - 1) + 2 = O(n)$ vertices and $|E(G)| = n + 5 \cdot (n - 1) + 3 = O(n)$ arcs. Hence G can be created in polynomial time.

The construction of G implies a bijection between solutions $\mathbf{x} \in \mathcal{X}$ and s - t paths P in G , such that $x_k = 1$ if and only if $(k_s, k_t) \in P$ for $k \in N$. The empty assortment maps to the single-edge path $P = (s, t)$. Furthermore the objective value of \mathbf{x} in Problem (2.1) is equal to the length of the path P under costs \mathbf{c}^γ .

As G is a DAG, it is well-known that the longest path can be computed via dynamic programming. Let $C_\gamma(u)$ be the cost of the longest s - u path in G under cost vector \mathbf{c}^γ . Solving Problem (2.1) for the optimal solution \mathbf{x} requires us to

compute the longest path P on G , which gives us $f(\gamma) = C_\gamma(t)$. Since $C_\gamma(t)$ already considers displaying the empty assortment via the (s, t) arc, this solution does not have to be computed separately as in Section 2.3. We solve the following dynamic program to compute $C_\gamma(t)$:

$$\begin{aligned} C_\gamma(s) &= 0, \\ C_\gamma(u) &= \max_{v: (v,u) \in E(G)} C_\gamma(v) + c_{v,u}^\gamma \quad \forall u \neq s. \end{aligned}$$

Computing $C_\gamma(t)$ involves computing $C_\gamma(u)$ at $O(n)$ states. There are at most two decisions to consider for each $C_\gamma(u)$, because each vertex $u \in V(G)$ has at most two incoming arcs. Hence, we can compute $C_\gamma(t)$ in $O(n)$ operations. Since we can compute $f(\gamma)$ efficiently, we can use Newton's method to search for γ^* such that $f(\gamma^*) = w_0 \gamma^*$ in $O(n^2 \log^2 n)$ iterations of solving $C_\gamma(t)$. Our total runtime remains $O(n^3 \log^2 n)$ in the showroom setting.

A.2.2 General Setting

Next, we move onto the problem of solving Problem (2.3) for $\tilde{f}(\gamma, R, U)$ when inputs have been rounded appropriately for any (R, U) pair. In order to incorporate the two constraints of Problem (2.3) into our longest path problem, we add two sets of weights, \mathbf{d}^1 and \mathbf{d}^2 , to our graph G and turn it into a constrained longest path problem. We will ensure that our path length is at least Y_1 under weights \mathbf{d}^1 and no longer than Y_2 under weights \mathbf{d}^2 . Due to the bijection between s - t paths in G and solutions $\mathbf{x} \in \mathcal{X}$, we know that $x_k = 1$ if and only if the arc (k_s, k_t) is in the path P for $k \in N$. The obvious way to construct \mathbf{d}^1 and \mathbf{d}^2 in

G_{root} are:

$$d_{u,v}^1 = \begin{cases} \tilde{\pi}_k^R & : (u, v) = (k_s, k_t), k \in N \\ 0 & : \text{otherwise} \end{cases},$$

$$d_{u,v}^2 = \begin{cases} \tilde{v}_k^U & : (u, v) = (k_s, k_t), k \in N \\ 0 & : \text{otherwise} \end{cases}.$$

To incorporate the weight of the no-purchase option, \tilde{v}_0^U , into the path length under \mathbf{d}^2 , we add weights to the three arcs in $E(G) \setminus E(G_{\text{root}})$:

$$d_{u,v}^1 = 0, \quad \text{for } (u, v) = (s, t), (s, \text{root}_s), (\text{root}_t, t),$$

$$d_{u,v}^2 = \begin{cases} \tilde{v}_0^U & : (u, v) = (\text{root}_t, t), (s, t) \\ 0 & : (u, v) = (s, \text{root}_s) \end{cases}.$$

An s - t path that ensures $d^1(P) \geq Y_1$ and $d^2(P) \leq Y_2$ corresponds to a solution \mathbf{x} such that the constraints of Problem (2.3) are satisfied. Since Y_1 and Y_2 are integers of the order $O(n/\epsilon)$, we can set up a dynamic program to compute the longest s - t path under costs \mathbf{c}^γ subject to the length requirements $d^1(P) \geq Y_1$ and $d^2(P) \leq Y_2$. Let $\tilde{C}_\gamma^{R,U}(u, y_1, y_2)$ denote the length of the longest s - u path P_u such that $d^1(P_u) \geq y_1$ and $d^2(P_u) \leq y_2$. Then $\tilde{f}(\gamma, R, U) = \tilde{C}_\gamma^{R,U}(t, Y_1, Y_2)$. Our dynamic program is as follows for $y_1 \in \{0, \dots, Y_1\}$ and $y_2 \in \{0, \dots, Y_2\}$:

$$\tilde{C}_\gamma^{R,U}(s, y_1, y_2) = \begin{cases} 0 & : y_1 \leq 0, y_2 \geq 0 \\ -\infty & : \text{otherwise} \end{cases},$$

$$\tilde{C}_\gamma^{R,U}(u, y_1, y_2) = \max_{v: (v,u) \in E(G)} \left\{ c_{v,u}^\gamma + \tilde{C}_\gamma^{R,U}(v, y_1 - d_{v,u}^1, y_2 - d_{v,u}^2) \right\} \quad \forall u \neq s.$$

When we compute $\tilde{C}_\gamma^{R,U}(u, y_1, y_2)$, we may move backward to a state where the second or third input is negative. This happens if $y_1 - d_{v,u}^1 < 0$ or $y_2 - d_{v,u}^2 < 0$.

In the former case, the state (u, y_1, y_2) with $y_1 < 0$ means that $d^1(P_u) < 0$ and our requirement under weights \mathbf{d}^1 has been satisfied. To reduce the number of states in our dynamic program, we can set $\tilde{C}_\gamma^{R,U}(u, y_1, y_2) = \tilde{C}_\gamma^{R,U}(u, 0, y_2)$ whenever $y_1 < 0$, because we do not have to consider weights \mathbf{d}^1 at either states.

In the latter case, the state (u, y_1, y_2) with $y_2 < 0$ means that $d^2(P_u) < 0$ and we violate our requirement under \mathbf{d}^2 . Furthermore, the base case ensures that $\tilde{C}_\gamma^{R,U}(u, y_1, y_2) = -\infty$ if we continue to run the dynamic program, regardless of the value of y_1 . Hence, we can set $\tilde{C}_\gamma^{R,U}(u, y_1, y_2) = -\infty$ whenever $y_2 < 0$ to reduce the number of states in our dynamic program.

One final difference between the path-based dynamic program and the original dynamic program is that we were computing $\tilde{V}_\gamma^{R,U}(\text{root}, Y_1, Y_2 - \tilde{v}_0^U)$, whereas we are now computing $\tilde{C}_\gamma^{R,U}(t, Y_1, Y_2)$. We do not need to adjust Y_2 by \tilde{v}_0^U in the path-based dynamic program because its value has been incorporated into G by arcs (s, t) and (root, t) .

Theorem 9. *Suppose Π^* is the optimal expected revenue of the OCA problem under the features tree model. For any $\epsilon \in (0, 1)$, there exists an algorithm that finds an in-store assortment \mathbf{x} such that $\Pi(\mathbf{x}) \geq (1 - \epsilon)\Pi^*$ in $O\left(\frac{n^5 \log^2 n \log n\bar{R}/R \cdot \log(n+1)\bar{U}/U}{\epsilon^4}\right)$ operations.*

Proof. The $(1 - \epsilon)$ -approximation follows from the proof of Theorem 2, where we take $\epsilon' = \epsilon/6$ when creating our grid \mathcal{K}_ϵ and rounding our values.

Also similar to the proof of Theorem 2, the number of times that we need to compute $\tilde{C}_\gamma^{R,U}(t, Y_1, Y_2)$ is $O\left(\frac{n^2 \log^2 n \log n\bar{R}/R \cdot \log(n+1)\bar{U}/U}{\epsilon^2}\right)$.

To compute $\tilde{C}_\gamma^{R,U}(t, Y_1, Y_2)$, there are $O(n)$ vertices in G and $O(n^2/\epsilon^2)$ pairs of (y_1, y_2) for each vertex. So there are $O(n^3/\epsilon^2)$ states. For each state, the dynamic program considers at most two decisions as there are at most two in-

coming arcs per vertex. Therefore, finding the longest constrained path in G takes $O(n^3/\epsilon^2)$ operations and the total runtime of the modified algorithm is $O(\frac{n^5 \log^2 n \log n \bar{R}/R \cdot \log(n+1) \bar{U}/U}{\epsilon^4})$ operations. \square

Compared to Theorem 2, the number of operations decreased by a factor of $O(n^2/\epsilon^2)$. This is because we reduce the number of decisions at each state from $O(n^2/\epsilon^2)$ to $O(1)$, while increasing the number of states by a factor of 2. When we run the original dynamic program in Subsection 2.4.2 and consider state (k, y_1, y_2) , we immediately decide on the allocation of (y_1, y_2) between the two children of k . This is wasteful because we do not consider the actual revenues and offline preference weights of products in the subtree, and different ways of splitting (y_1, y_2) could have the same results. Using the new dynamic program, we only decide on displaying one or both children features of k when we are at (k, y_1, y_2) , not how we split (y_1, y_2) over each subtree. The actual allocation of (y_1, y_2) is delayed until we reach the parent of a product vertex, when we decide whether to offer the left, right, or both products. The graph G sorts vertices in the features tree according to a depth-first-search, which allows us to consider the consumption of budgets (y_1, y_2) product by product, rather than subtree by subtree like the original dynamic program.

This constrained longest path construction can be used to represent the parametrized version of Problem (2.2) at (R, U) before rounding parameters $\pi_i \hat{v}_i$ and \hat{v}_i . We could have simply used $\pi_i \hat{v}_i$ and \hat{v}_i in the construction of \mathbf{d}^1 and \mathbf{d}^2 , replacing the path constraints with $d^1(P) \geq R$ and $d^2(P) \leq U$. We present the rounded version directly to show how we can compute a solution in polynomial time.

With regards to the extensions in Section 2.6, where the retailer chooses both

online and in-store assortments, the earlier modifications can be incorporated by applying the non-negativity function to arc costs whenever the corresponding terms are non-negative in the earlier dynamic program.

A.3 Ground-Truth and Benchmark Models

In this section, we give the details underlying the ground-truth and benchmark models, against which we test our features tree model.

A.3.1 Ground-Truth Model

Our ground-truth model for testing the modeling power of our features tree model is a generalization of the model introduced by Dzyabura and Jagabathula (2017). We do not require the online store to offer a product for every possible combination of features. For L feature classes and K feature values per class, our ground-truth model selects $n \leq L^K$ products as the full assortment.

As a result of this modification, our ground-truth model differs from Dzyabura and Jagabathula (2017) because it cannot be optimized over features. Rather, we require a separate set of variables to describe which products and features are on display. Product i has initial preference weight w_i and its preference weight is updated when the features of product i are seen in-store. We use binary variables $\mathbf{x} \in \{0, 1\}^n$ to describe whether products are on display, so that $x_i = 1$ if product i is on display and 0 otherwise.

We denote features by their class-value pair (ℓ, k) . We use $\mathbf{y} \in \{0, 1\}^{L \cdot K}$ to

indicate which features are seen. Each feature (ℓ, k) has a boost or discount multiplier $\delta_{\ell,k}$. To be consistent with notation in our features tree model, let $A(i)$ denote the features of product i and $L(\ell, k)$ denote all the products with feature (ℓ, k) . Following the same logic in Section 2.2, we have $y_{\ell,k} = 1$ if and only if $x_i = 1$ for some $i \in L(\ell, k)$, and the set of feasible characteristic vectors (\mathbf{x}, \mathbf{y}) describing the features and products on display can be denoted by \mathcal{X}^G , where:

$$\mathcal{X}^G = \left\{ (\mathbf{x}, \mathbf{y}) \left| \begin{array}{ll} x_i \leq y_{\ell,k} & \forall i, (\ell, k) \in A(i), \\ y_{\ell,k} \leq \sum_{i \in L(\ell,k)} x_i & \forall (\ell, k), \\ y_{\ell,k} \in \{0, 1\} & \forall (\ell, k), \\ x_i \in \{0, 1\} & \forall i. \end{array} \right. \right\}.$$

Given an assortment \mathbf{x} , the vector \mathbf{y} is uniquely defined. The final preference weight of product i can be written as $w_i \cdot \prod_{(\ell,k) \in A(i)} \delta_{\ell,k}^{y_{\ell,k}}$ and the probability of selling product i in assortment \mathbf{x} is:

$$P_i^G(\mathbf{x}, \mathbf{y}) = \frac{w_i \cdot \prod_{(\ell,k) \in A(i)} \delta_{\ell,k}^{y_{\ell,k}}}{w_0 + \sum_{j=1}^n w_j \cdot \prod_{(\ell,k) \in A(j)} \delta_{\ell,k}^{y_{\ell,k}}}.$$

Hence the assortment optimization problem in the showroom setting under the ground-truth model is to maximize expected revenue $\Pi^G(\mathbf{x}, \mathbf{y})$ over \mathcal{X}^G , where where $\Pi^G(\mathbf{x}, \mathbf{y})$ is defined as:

$$\Pi^G(\mathbf{x}, \mathbf{y}) = \frac{\sum_{i=1}^n \pi_i w_i \cdot \prod_{(\ell,k) \in A(i)} \delta_{\ell,k}^{y_{\ell,k}}}{w_0 + \sum_{i=1}^n w_i \cdot \prod_{(\ell,k) \in A(i)} \delta_{\ell,k}^{y_{\ell,k}}}.$$

A.3.2 Benchmark Model

Our benchmark model is the simplest model that incorporates the notion of having different preference weights for products, depending on whether they are on display. In the benchmark model, we ignore features relationship and product i simply has a preference weight of v_i when it is displayed and w_i otherwise.

We use binary variables $\mathbf{x} \in \{0, 1\}^n$ to describe the assortment being display in-store. That is, $x_i = 1$ if product i is on display and 0 otherwise. The preference weight of product i is $v_i x_i + w_i \cdot (1 - x_i)$ and the probability of selling product i when assortment \mathbf{x} is offered is:

$$P_i^B(\mathbf{x}) = \frac{v_i x_i + w_i \cdot (1 - x_i)}{w_0 + \sum_{j=1}^n v_j x_j + w_j \cdot (1 - x_j)}.$$

Hence the assortment optimization problem in the showroom setting under the benchmark model is to maximize expected revenue $\Pi^B(\mathbf{x})$ over $\{0, 1\}^n$, where $\Pi^B(\mathbf{x})$ is defined as:

$$\Pi^B(\mathbf{x}) = \frac{\sum_{i=1}^n \pi_i (v_i x_i + w_i \cdot (1 - x_i))}{w_0 + \sum_{i=1}^n v_i x_i + w_i \cdot (1 - x_i)}.$$

APPENDIX B

APPENDIX: ASSORTMENT OPTIMIZATION UNDER THE MULTINOMIAL LOGIT MODEL WITH PRODUCT SYNERGIES

B.1 Graphs with Low Treewidth

We present a dynamic program to solve the parametrized problem, Problem (3.2), when we are given a tree decomposition of the synergy graph with width t . For a synergy graph $G = (V, E)$, let (X, T) denote a tree decomposition of G , where $T = (V', E')$ is a tree. Furthermore, $X = \{X_1, \dots, X_K\}$ is a collection of subsets of vertices in V , such that each vertex $k \in V'$ is associated with $X_k \subseteq V$, and X satisfies three properties:

1. Every vertex of G is in some X_k : $\cup_{k \in V'} X_k = V$.
2. For every $(i, j) \in E$, there exists $k \in V'$ such that $i, j \in X_k$.
3. Suppose $m \in V'$ is on the unique k - ℓ path in T . Then $X_k \cap X_\ell \subseteq X_m$.

The width of (X, T) is defined as $t = \max_{k \in V'} |X_k| - 1$, and the treewidth t^* of G is the minimum width of all its tree decompositions.

Select a vertex of V' to be the root of T . Since we work with subsets of vertices, we use set notation rather than vector notation in this appendix. We use p_k to denote the parent of k on T and C'_k to denote the set of children of k . Let T_k denote the set of products which are in some X_ℓ , where ℓ is a vertex in the subtree rooted at k . Recursively, $T_k = X_k$ if k is a leaf and $T_k = (\cup_{c \in C'_k} T_c) \cup X_k$ if k is not a leaf. Notice that $C'_k \subseteq V'$ and $X_k \subseteq T_k \subseteq V$, so they are subsets of vertices on different graphs.

Define $R^\delta(S)$ as the contribution to Problem (3.2) from the vertices and edges in the subgraph of G induced by products in S . In other words,

$$R^\delta(S) = \sum_{i=1}^n (\pi_i - \delta) u_i \cdot \mathbb{1}[i \in S] \\ + \sum_{i=1}^{n-1} \sum_{j=i+1}^n (\tilde{\pi}_{i,j} - \delta) v_{i,j} \cdot \mathbb{1}[i, j \in S].$$

Our value function is defined on the tree rather than the original graph. At a vertex $k \in V'$, our value function considers the maximum contribution to Problem (3.2) that we can achieve by offering products within $S \subseteq T_k$, given that we include $S_{p_k} \subseteq X_{p_k}$ in the assortment. A tree decomposition allows product i to be in X_{p_k} and T_k , so we must ensure that such a product is included in S if and only if it is included in S_{p_k} . If so, we say that $S \subseteq T_k$ is feasible with respect to S_{p_k} . More precisely, given $S_{p_k} \subseteq X_{p_k}$, the feasible subsets of T_k are:

$$F_k^T(S_{p_k}) = \{S \subseteq T_k : S \cap X_{p_k} = S_{p_k} \cap T_k\}.$$

The value function at $k \in V'$ is:

$$V_k(S_{p_k}) = \max_{S \in F_k^T(S_{p_k})} R^\delta(S).$$

To formulate a dynamic program, we want to restrict our current decision to $S \subseteq X_k$ rather than $S \subseteq T_k$. Similar to the definition of $F_k^T(S_{p_k})$, we want to define the feasible subsets that we may offer from X_k when we are given S_{p_k} . Let $F_k^X(S_{p_k})$ be the collection of feasible subsets that we may offer:

$$F_k^X(S_{p_k}) = \{S \subseteq X_k : S \cap X_{p_k} = S_{p_k} \cap X_k\}.$$

The next lemma proves we can compute our value function via dynamic programming using the definition of $F_k^X(S_{p_k})$ and $R^\delta(S)$.

Lemma 8. *The value function $V_k(S_{p_k})$ can be written as:*

$$V_k(S_{p_k}) = \max_{S_k \in F_k^X(S_{p_k})} R^\delta(S_k) + \sum_{c \in C'_k} V_c(S_k) - R^\delta(S_k \cap X_c).$$

The intuition behind the dynamic program is that the decision made at vertex ℓ , where ℓ is in the subtree rooted at k , is feasible with respect to S_{p_k} . This is because a product $i \in X_{p_k} \cap X_\ell$ must also be in X_k , as well as all X_m where m is on the k - ℓ path in the tree. Hence we would make the same decision regarding product i at each tree vertex from k to ℓ . To prove the correctness of the dynamic program, we check that we do not double-count a product or the synergy between products since we may observe a graph vertex or edge in multiple X_ℓ . Furthermore, if $(i, j) \in E$ and $i, j \in T_k$, we check that both i, j are contained in some X_ℓ to ensure that their synergy contribution is recorded. We first use Lemma 8 to prove the runtime analysis in Theorem 4, and then revisit Lemma 8 to prove the correctness of the dynamic program.

Proof of Theorem 4. For each vertex on the tree, the state space is the collection of subsets of X_{p_k} and has size $O(2^t)$ since $|X_{p_k}| \leq t - 1$. We consider including $S \in F_k^X(S_{p_k})$ in the assortment, so we consider at most $O(2^t)$ decisions. Our dynamic program is computed from the leaves of T up to the root, and the tree has $O(n)$ vertices. Hence Problem (3.2) can be solved in $O(2^t n)$ operations.

If we use Newton's algorithm to find the fixed point δ^* , then we need to solve Problem (3.2) at most $O(n^4 \log^2 n)$ times. Hence the total number of operations to find the optimal assortment is $O(2^t n^5 \log^2 n)$. \square

In order to prove Lemma 8, we use a series of claims to show that the decisions for products in different subtrees T_c are independent when we are given

the decision to include $S_k \subseteq X_k$, where c is a child of k . We first show that a product in T_k must either be in X_k or in exactly one T_c so that we can split the decision over vertex k and its subtrees. Similarly, a pair of products with positive synergy must either both be in X_k or both be in exactly one T_c .

Claim 3. *Consider vertex $k \in V'$ and two of its children $c, c' \in C'_k$. Then no products are in both T_c and $T_{c'}$ unless they are also in X_k : $(T_c \setminus X_k) \cap (T_{c'} \setminus X_k) = \emptyset$.*

Proof. Suppose by contradiction that there exists some product i such that $i \in (T_c \setminus X_k) \cap (T_{c'} \setminus X_k)$. Then there exists some $\ell \in V'$ in the subtree rooted at c , such that $i \in X_\ell$. Similarly, there exists $\ell' \in V'$ in the subtree rooted at c' such that $i \in X_{\ell'}$. The path from ℓ to ℓ' contains k , so $X_\ell \cap X_{\ell'} \subseteq X_k$. This implies $i \in X_k$ and contradicts $i \in (T_c \setminus X_k) \cap (T_{c'} \setminus X_k)$. \square

Claim 4. *Suppose there exists an edge $(i, j) \in E$ such that $i, j \in T_k$. Then either $i, j \in X_k$, or at least one of products i, j is not in X_k and there exists a unique $c \in C'_k$ such that $i, j \in T_c$.*

Proof. By definition of tree decomposition, there exists some $\ell \in V'$ such that $i, j \in X_\ell$. If $\ell = k$, then the first condition is satisfied. If $\ell \neq k$ but is a vertex in the subtree rooted at k , then ℓ must be in the subtree rooted at a vertex $c \in C'_k$, and $i, j \in T_c$. Furthermore, the uniqueness of c follows from observing that at least one of i, j is not in X_k and applying Claim 3. This satisfies the second condition. We show that such a vertex ℓ must exist in the subtree rooted at k .

Suppose by contradiction that ℓ is not a vertex in the subtree rooted at k . There must exist vertices ℓ', ℓ'' in the subtree rooted at k , $\ell' \neq \ell''$, such that $i \in X_{\ell'} \setminus X_{\ell''}$ and $j \in X_{\ell''} \setminus X_{\ell'}$ since $i, j \in T_k$. The unique path from ℓ to ℓ' on T contains k , and similarly the path from ℓ to ℓ'' contains k . This implies that

$X_\ell \cap X_{\ell'} \subseteq X_k$, so $i \in X_k$. Similarly, $X_\ell \cap X_{\ell''} \subseteq X_k$, so $j \in X_k$. This contradicts our assumption that no vertex ℓ exists in the subtree rooted at k such that $i, j \in X_\ell$. \square

The next claim breaks up $R^\delta(S)$ over X_k and T_c for $c \in C'_k$, so that we can count the contributions from X_k and T_c separately.

Lemma 9. *At vertex $k \in V'$, suppose we have $S \subseteq T_k$. Then*

$$R^\delta(S) = R^\delta(S \cap X_k) + \sum_{c \in C'_k} R^\delta(S \cap T_c) - R^\delta(S \cap T_c \cap X_k).$$

Proof. By definition of $R^\delta(S)$, we have

$$\begin{aligned} R^\delta(S) &= \sum_{i=1}^n (\pi_i - \delta) u_i \cdot \mathbb{1}[i \in S] \\ &\quad + \sum_{i=1}^{n-1} \sum_{j=i+1}^n (\bar{\pi}_{i,j} - \delta) v_{i,j} \cdot \mathbb{1}[i, j \in S]. \end{aligned}$$

First, we consider the sum over the vertices of G . For a fixed product $i \in T_k$, we have,

$$\begin{aligned} \mathbb{1}[i \in S] &= \mathbb{1}[i \in S \cap X_k] \\ &\quad + \sum_{c \in C'_k} \mathbb{1}[i \in S \cap T_c] - \mathbb{1}[i \in S \cap T_c \cap X_k]. \end{aligned} \tag{B.1}$$

If $i \in X_k$, then $\mathbb{1}[i \in S \cap T_c] = \mathbb{1}[i \in S \cap T_c \cap X_k]$, so we count the contribution of product i when it appears in X_k . If $i \notin X_k$, then Claim 3 tells us that there exists an unique $c \in C'_k$ such that $i \in T_c$.

Next, we consider the sum over the edges of G . Suppose $(i, j) \in E$ so that $v_{i,j} \neq 0$, then the following is true:

$$\begin{aligned} \mathbb{1}[i, j \in S] &= \mathbb{1}[i, j \in S \cap X_k] \\ &\quad + \sum_{c \in C'_k} \mathbb{1}[i, j \in S \cap T_c] - \mathbb{1}[i, j \in S \cap T_c \cap X_k]. \end{aligned} \tag{B.2}$$

If $i, j \in X_k$, then $\mathbb{1}[i, j \in S \cap T_c] = \mathbb{1}[i, j \in S \cap T_c \cap X_k]$ and we count the synergy between the two products in X_k . Otherwise, Claim 4 implies that there is a unique c such that $i, j \in T_c$.

We can expand $R^\delta(S)$ and apply equalities (B.1) and (B.2). Furthermore, recall that $v_{i,j} = 0$ if $(i, j) \notin E$, so equality (B.2) holds true for all pairs i, j if we multiply both sides by $v_{i,j}$. Grouping the terms by their indicator functions, we see that $R^\delta(S) = R^\delta(S \cap X_k) + \sum_{c \in C'_k} R^\delta(S \cap T_c) - R(S \cap T_c \cap X_k)$. \square

Finally, we prove that every subset in $F_k^T(S_{p_k})$ can be written as the union of $S_k \in F_k^X(S_{p_k})$ and $S_c \in F_c^T(S_k)$ for $c \in C'_k$, and vice versa.

Claim 5. *Suppose we are given the decision $S_{p_k} \subseteq X_{p_k}$ and consider a subset of products $S \subseteq F_k^T(S_{p_k})$. Define $S_k = S \cap X_k$ and $S_c = S \cap T_c$ for $c \in C'_k$. Then $S_k \in F_k^X(S_{p_k})$ and $S_c \in F_c^T(S_k)$.*

Proof. Since $S \in F_k^T(S_{p_k})$, we know that

$$S \cap X_{p_k} = S_{p_k} \cap T_k.$$

We intersect both sides with X_k to obtain:

$$S \cap X_{p_k} \cap X_k = S_{p_k} \cap T_k \cap X_k.$$

Since $S_k = S \cap X_k$ and $X_k \subseteq T_k$, we have $S_k \cap X_{p_k} = S_{p_k} \cap X_k$. Hence $S_k \in F_k^X(S_{p_k})$.

Finally, to show that $S_c \in F_c^T(S_k)$, observe:

$$S_c \cap X_k = S \cap T_c \cap X_k = S_k \cap T_c.$$

\square

Claim 6. *Given $S_{p_k} \subseteq X_{p_k}$, suppose we have $S_k \in F_k^X(S_{p_k})$ and $S_c \in F_c^T(S_k)$ for all $c \in C'_k$. Let S be the union of these sets such that $S = (\cup_{c \in C'_k} S_c) \cup S_k$, then $S \in F_k^T(S_{p_k})$.*

Proof. We consider:

$$\begin{aligned} S \cap X_{p_k} &= (S_k \cap X_{p_k}) \\ &\cup \left(\bigcup_{c \in C'_k} (S_c \cap X_{p_k}) \right). \end{aligned} \tag{B.3}$$

First we show that $X_{p_k} \cap T_k = X_{p_k} \cap X_k$. The “ \supseteq ” direction is due to $T_k \supseteq X_k$. For the “ \subseteq ” direction, observe that for every $\ell \in V'$ in the subtree rooted at k , including $\ell = k$, we have $X_{p_k} \cap X_\ell \subseteq X_k$ by the third property of (X, T) being a tree decomposition. Since T_k is the union of all X_ℓ , we have $X_{p_k} \cap T_k \subseteq X_k$. Hence $X_{p_k} \cap T_k = X_{p_k} \cap X_k$, which also implies that $S_{p_k} \cap T_k = S_{p_k} \cap X_k$.

For the first term in the right side of equality (B.3), we use $S_k \in F_k^X(S_{p_k})$ to obtain:

$$\begin{aligned} S_k \cap X_{p_k} &= S_{p_k} \cap X_k \\ &= S_{p_k} \cap T_k. \end{aligned}$$

For the second term, consider some $c \in C'_k$. We have:

$$\begin{aligned} S_c \cap X_{p_k} &= S_c \cap X_{p_k} \cap X_k \\ &= S_k \cap T_c \cap X_{p_k} \\ &= S_p \cap X_k \cap T_c. \end{aligned}$$

The first line is true since $S_c \cap X_{p_k} \subseteq T_k \cap X_{p_k} \subseteq X_k$. The second line uses $S_c \in F_c^T(S_k)$, and the third line uses $S_k \in F_k^X(S_{p_k})$. Finally, this second term is a subset of the first term, so:

$$S \cap X_{p_k} = S_{p_k} \cap T_k,$$

and hence $S \in F_k^T(S_{p_k})$ as required. \square

We are now ready to prove Lemma 8.

Proof of Lemma 8. We prove the correctness of our dynamic program via induction. In the base case of a leaf, the dynamic program is correct because a leaf vertex does not have children.

At a non-leaf vertex $k \in V'$, assume that the dynamic program holds for all descendants in the subtree rooted at k on T . Apply Lemma 9:

$$\begin{aligned} V_k(S_{p_k}) &= \max_{S \in F_k^T(S_{p_k})} R^\delta(S) \\ &= \max_{S \in F_k^T(S_{p_k})} R^\delta(S \cap X_k) \\ &\quad + \sum_{c \in C'_k} R^\delta(S \cap T_c) - R^\delta(S \cap T_c \cap X_k). \end{aligned}$$

Claims 5 and 6 state that we can choose S_k from $F_k^X(S_{p_k})$ and then S_c from $F_c^T(S_k)$ such that $S = (\cup_{c \in C'_k} S_c) \cup S_k$, rather than S from $F_k^T(S_{p_k})$. This gives us:

$$\begin{aligned} V_k(S_{p_k}) &= \max_{S_k \in F_k^X(S_{p_k})} \left(R^\delta(S_k) \right. \\ &\quad \left. + \sum_{c \in C'_k} \max_{S_c \in F_c^T(S_k)} \left(R^\delta(S_c) - R^\delta(S_c \cap X_k) \right) \right) \\ &= \max_{S_k \in F_k^X(S_{p_k})} \left(R^\delta(S_k) \right. \\ &\quad \left. + \sum_{c \in C'_k} \max_{S_c \in F_c^T(S_k)} R^\delta(S_c) - R^\delta(S_k \cap X_c) \right) \\ &= \max_{S_k \in F_k^X(S_{p_k})} R^\delta(S_k) + \sum_{c \in C'_k} V_c(S_k) - R^\delta(S_k \cap X_c). \end{aligned}$$

The second line is true because $S_c \in F_c^T(S_k)$. The last line is true by observing that only the second term is dependent on the inner maximization, and applying the induction hypothesis to get $V_c(S_k) = \max_{S_c \in F_c^T(S_k)} R^\delta(S_c)$. \square

B.2 Constructing the Sales-based Linear Program

In this section, we construct the SBLP in Problem (3.7), and show that the extreme points of its feasible region satisfy condition 1. As a result, we can compute the optimal assortment simply by solving a linear program, rather than going through multiple iterations of a dynamic program.

Problem (3.6) expands out to the following linear program when we implicitly set $\delta = V_1$:

$$\begin{aligned}
& \min \quad V_1 \\
& \text{s.t.} \quad V_1 \geq V_2^0 \\
& \quad V_1 \geq (\pi_1 - V_1)u_1 + V_2^1 \\
& \quad V_i^0 \geq V_{i+1}^0 \quad \forall i = 2, \dots, n \\
& \quad V_i^0 \geq (\pi_i - V_1)u_i + V_{i+1}^1 \quad \forall i = 2, \dots, n \\
& \quad V_i^1 \geq V_{i+1}^0 \quad \forall i = 2, \dots, n \\
& \quad V_i^1 \geq (\bar{\pi}_{i-1} - V_1)v_{i-1} + (\pi_i - V_1)u_i + V_{i+1}^1 \quad \forall i = 2, \dots, n \\
& \quad V_{n+1}^0 = V_{n+1}^1 = 0.
\end{aligned}$$

By Lemma 3, the above linear program has optimal objective value δ^* . Substitute in the value of $V_{n+1}^0 = V_{n+1}^1 = 0$ and associate the following dual variables with the constraints relating products $i - 1$ and i :

- z_i^{00} when we do not offer $i - 1$ and i ,
- z_i^{01} when we do not offer $i - 1$ but offer i ,
- z_i^{10} when we offer $i - 1$ but do not offer i , and
- z_i^{11} when we offer both $i - 1$ and i .

We define the above for $i = 1, \dots, n$ with $z_1^{10} = z_1^{11} = 0$, because there is no profit nor synergy associated with the no-purchase option. The dual is:

$$\begin{aligned}
& \max \sum_{i=1}^n \pi_i u_i (z_i^{01} + z_i^{11}) + \sum_{i=1}^{n-1} \bar{\pi}_i v_i z_{i+1}^{11} \\
& \text{s.t. } z_1^{00} + z_1^{01} + \sum_{i=1}^n u_i (z_i^{01} + z_i^{11}) + \sum_{i=1}^{n-1} v_i z_{i+1}^{11} = 1 \\
& \quad z_i^{00} - z_{i-1}^{00} + z_i^{01} - z_{i-1}^{10} = 0 \quad \forall i = 2, \dots, n \\
& \quad z_i^{10} - z_{i-1}^{01} + z_i^{11} - z_{i-1}^{11} = 0 \quad \forall i = 2, \dots, n \\
& \quad z_i^{00}, z_i^{01}, z_i^{10}, z_i^{11} \geq 0 \quad \forall i = 1, \dots, n \\
& \quad z_1^{10} = z_1^{11} = 0.
\end{aligned}$$

Lemma 10. *The dual to Problem (3.6) is equivalent to Problem (3.7).*

Proof. Suppose we have a solution \mathbf{z} to the dual of Problem (3.6), then we can construct our solution (\mathbf{y}, \mathbf{w}) as:

$$\begin{aligned}
y_0 &= z_1^{00} + z_1^{01}, \\
y_i &= u_i (z_i^{01} + z_i^{11}), \quad \forall i = 1, \dots, n, \\
w_i &= v_i z_{i+1}^{11}, \quad \forall i = 1, \dots, n-1.
\end{aligned}$$

It is simple to verify that (\mathbf{y}, \mathbf{w}) is feasible to Problem (3.7) and has the same objective value as \mathbf{z} .

In the reverse direction, if we have a solution (\mathbf{y}, \mathbf{w}) to Problem (3.7), we can

construct a solution \mathbf{z} as follows:

$$\begin{aligned}
z_1^{00} &= x_0 - \frac{y_1}{u_1}, \\
z_i^{00} &= x_0 - \frac{y_i}{u_i} - \frac{y_{i-1}}{u_{i-1}} + \frac{w_{i-1}}{v_{i-1}}, & \forall i = 2, \dots, n, \\
z_1^{01} &= \frac{y_1}{u_1}, \\
z_i^{01} &= \frac{y_i}{u_i} - \frac{w_{i-1}}{v_{i-1}}, & \forall i = 2, \dots, n, \\
z_i^{10} &= \frac{y_{i-1}}{u_{i-1}} - \frac{w_{i-1}}{v_{i-1}}, & \forall i = 2, \dots, n, \\
z_i^{11} &= \frac{w_{i-1}}{v_{i-1}}, & \forall i = 2, \dots, n, \\
z_1^{10} &= z_1^{11} = 0.
\end{aligned}$$

This solution \mathbf{z} is feasible to the dual and has the same objective value as (\mathbf{y}, \mathbf{w}) in Problem (3.7). \square

The key to proving Theorem 5 is to show that every feasible point with some y_i such that $0 < y_i/u_i < y_0$ can be written as a convex combination of two other feasible points, and hence it is not an extreme point.

Lemma 11. *Every extreme point to the feasible region in Problem (3.7) satisfies $y_i = 0$ or $y_i/u_i = y_0$ for all $i = 1, \dots, n$.*

Proof. Suppose we have a solution (\mathbf{y}, \mathbf{w}) with at least one variable y_i such that $0 < y_i/u_i < y_0$. Define the set of non-tight variables as $J = \{y_i : 0 < y_i/u_i < y_0\} \cup \{w_i : 0 < w_i/v_i < y_0\}$.

Order the variables as $y_1, w_1, y_2, \dots, y_{n-1}, w_{n-1}, y_n$. Define the set of indices $B = \{b_1, \dots, b_{L+1}\}$ based on the tightness of the third constraint of Problem (3.7).

Specifically, $b_1 = 0$, $b_{L+1} = n$, $b_1 < b_2 < \dots < b_{L+1}$, and $i \in B$ if and only if $y_0 = y_i/u_i + y_{i+1}/u_{i+1} - w_i/v_i$.

We partition the variables into $L + 1$ sets, A_0, A_1, \dots, A_L . For A_ℓ such that $1 \leq \ell \leq L$, we define:

$$\begin{aligned} A_\ell = & \{y_i : b_\ell < i \leq b_{\ell+1}\} \cup \{w_i : b_\ell < i < b_{\ell+1}\} \\ & \cup \{w_i : i = b_{\ell+1} \neq n, w_i/v_i = y_i/u_i < y_{i+1}/u_{i+1}\} \\ & \cup \{w_i : i = b_\ell \neq 0, w_i/v_i = y_{i+1}/u_{i+1} < y_i/u_i\}. \end{aligned}$$

Set A_0 contains the remaining indices of \mathbf{w} :

$$\begin{aligned} A_0 = & \{w_i : i \in B, w_i/v_i = y_i/u_i = y_{i+1}/u_{i+1}\} \\ & \cup \{w_i : i \in B, w_i/v_i < \min\{y_i/u_i, y_{i+1}/u_{i+1}\}\}. \end{aligned}$$

To create a new solution, we first define a scaling factor β as:

$$\beta = \sum_{\substack{i: y_i \in J \cap A_\ell \\ \ell \text{ odd}}} u_i + \sum_{\substack{i: w_i \in J \cap A_\ell \\ \ell \text{ odd}}} v_i - \sum_{\substack{i: y_i \in J \cap A_\ell \\ \ell \neq 0 \text{ even}}} u_i - \sum_{\substack{i: w_i \in J \cap A_\ell \\ \ell \neq 0 \text{ even}}} v_i.$$

The value of β can any real number. For some small ϵ satisfying $0 < \epsilon < 1/\beta$, the new solution $(\bar{\mathbf{y}}, \bar{\mathbf{w}})$ is:

$$\begin{aligned} \bar{y}_i = & \begin{cases} \frac{1}{1+\epsilon\beta} y_i & \text{if } y_i \notin J \\ \frac{1}{1+\epsilon\beta} (y_i + \epsilon u_i) & \text{if } y_i \in J \cap A_\ell \text{ for odd } \ell \\ \frac{1}{1+\epsilon\beta} (y_i - \epsilon u_i) & \text{if } y_i \in J \cap A_\ell \text{ for even } \ell \neq 0 \end{cases}, \\ \bar{w}_i = & \begin{cases} \frac{1}{1+\epsilon\beta} w_i & \text{if } w_i \notin J \text{ or } w_i \in A_0 \\ \frac{1}{1+\epsilon\beta} (w_i + \epsilon v_i) & \text{if } w_i \in J \cap A_\ell \text{ for odd } \ell \\ \frac{1}{1+\epsilon\beta} (w_i - \epsilon v_i) & \text{if } w_i \in J \cap A_\ell \text{ for even } \ell \neq 0 \end{cases}, \\ \bar{y}_0 = & y_0/(1 + \epsilon\beta). \end{aligned}$$

It is clear that $\bar{y}_0 + \sum_{i=1}^n \bar{y}_i + \sum_{i=1}^{n-1} \bar{w}_i = 1$, so we need to check the other constraints.

Case 1 - $\bar{y}_i, \bar{w}_i, \bar{y}_{i+1} \in A_\ell$ for odd ℓ : If $w_i \notin J$, then $\bar{w}_i = w_i/(1 + \epsilon\beta)$ and $\bar{y}_i \geq y_i/(1 + \epsilon\beta)$, so we have $\bar{y}_i/u_i \geq \bar{w}_i/v_i$. If $w_i \in J$ and $y_i \in J$, then

$$\begin{aligned}\bar{y}_i/u_i &= (y_i/u_i + \epsilon)/(1 + \epsilon\beta) \\ &\geq (w_i/v_i + \epsilon)/(1 + \epsilon\beta) = \bar{w}_i/v_i.\end{aligned}$$

If $w_i \in J$ and $y_i \notin J$, then $y_i/u_i = y_0$, which implies $y_i/u_i > w_i/v_i$. There exists small ϵ such that:

$$\begin{aligned}\bar{y}_i/u_i &= y_i/u_i/(1 + \epsilon\beta) \\ &\geq (w_i/v_i + \epsilon)/(1 + \epsilon\beta) = \bar{w}_i/v_i.\end{aligned}$$

Using the same argument, we can show that $\bar{y}_{i+1}/\bar{u}_{i+1} \geq \bar{w}_i/\bar{v}_i$. Finally, observe:

$$\begin{aligned}&\frac{\bar{y}_i}{u_i} + \frac{\bar{y}_{i+1}}{u_{i+1}} - \frac{\bar{w}_i}{v_i} \\ &\leq \frac{1}{1 + \epsilon\beta} \left(\frac{y_i + \epsilon u_i}{u_i} + \frac{y_{i+1} + \epsilon u_{i+1}}{u_{i+1}} - \frac{\bar{w}_i}{v_i} \right) \\ &\leq \frac{y_0}{1 + \epsilon\beta} = \bar{y}_0,\end{aligned}$$

where the last inequality is true for small enough ϵ because the third constraint of Problem (3.7) is not tight, by definition of B .

Case 2 - $\bar{y}_i, \bar{w}_i, \bar{y}_{i+1} \in A_\ell$ for even $\ell \neq 0$: If $w_i \in J$, then

$$\begin{aligned}\bar{y}_i/u_i &\geq (y_i/u_i - \epsilon)/(1 + \epsilon\beta) \\ &\geq (w_i/v_i - \epsilon)/(1 + \epsilon\beta) = \bar{w}_i/v_i.\end{aligned}$$

If $w_i \notin J$ and $y_i \notin J$, then $\bar{w}_i = w_i/(1 + \epsilon\beta)$ and $\bar{y}_i = y_i/(1 + \epsilon\beta)$, so we have $\bar{y}_i/u_i \geq \bar{w}_i/v_i$. If $w_i \notin J$ and $y_i \in J$, we must have $w_i = 0$ and $y_i > 0$, so there exists small ϵ such that

$$\bar{y}_i/u_i = (y_i/u_i - \epsilon)/(1 + \epsilon\beta) \geq 0 = \bar{w}_i/v_i.$$

Finally,

$$\begin{aligned}
& \frac{\bar{y}_i}{u_i} + \frac{\bar{y}_{i+1}}{u_{i+1}} - \frac{\bar{w}_i}{u_i} \\
& \leq \frac{1}{1 + \epsilon\beta} \left(\frac{y_i}{u_i} + \frac{y_{i+1}}{u_{i+1}} - \frac{w_i - \epsilon v_i}{v_i} \right) \\
& \leq \frac{y_0}{1 + \epsilon\beta} = \bar{y}_0,
\end{aligned}$$

where the last equality again holds for small enough ϵ because the third constraint of Problem (3.7) is not tight, by definition of B .

Case 3 - $\bar{y}_i \in A_\ell$ and $\bar{y}_{i+1} \in A_{\ell+1}$ for odd ℓ : First suppose $w_i \in A_0$. If $w_i/v_i = y_i/u_i = y_{i+1}/u_{i+1}$, then they are equal to y_0 because the third constraint of Problem (3.7) is tight when y_i, y_{i+1} are in different sets. Hence they are all not in J and only scaled, so $\bar{w}_i/v_i = \bar{y}_i/u_i = \bar{y}_{i+1}/u_{i+1}$ and all the constraints still hold with equality. Otherwise, $w_i/v_i < \min\{y_i/u_i, y_{i+1}/u_{i+1}\}$, so there is some small ϵ such that $w_i/v_i \leq \min\{y_i/u_i, y_{i+1}/u_{i+1} - \epsilon\}$, which ensures $\bar{w}_i/v_i \leq \min\{\bar{y}_i/u_i, \bar{y}_{i+1}/u_{i+1}\}$. For the third constraint, notice that $y_0 - y_i/u_i = y_{i+1}/u_{i+1} - w_i/v_i > 0$ implies $y_i \in J$. Similarly, $y_{i+1} \in J$. Hence the following equalities hold:

$$\begin{aligned}
\bar{y}_0 &= \frac{y_0}{1 + \epsilon\beta} = \frac{1}{1 + \epsilon\beta} \left(\frac{y_i}{u_i} + \frac{y_{i+1}}{u_{i+1}} - \frac{w_i}{u_i} \right) \\
&= \frac{1}{1 + \epsilon\beta} \left(\frac{y_i + \epsilon u_i}{u_i} + \frac{y_{i+1} - \epsilon u_{i+1}}{u_{i+1}} - \frac{w_i}{u_i} \right) \\
&= \frac{\bar{y}_i}{u_i} + \frac{\bar{y}_{i+1}}{u_{i+1}} - \frac{\bar{w}_i}{u_i}.
\end{aligned}$$

Next, suppose $w_i \in A_\ell$. Then $y_i/u_i = w_i/v_i < y_{i+1}/u_{i+1}$, so $y_0 = y_{i+1}/u_{i+1}$ by tightness of the third constraint and $y_{i+1} \notin J$. Both w_i and y_i increase in the same direction, so $\bar{w}_i/v_i = \bar{y}_i/u_i$, and there exists small ϵ such that $\bar{w}_i/v_i \leq \bar{y}_{i+1}/u_{i+1}$. For the third constraint, \bar{y}_i and \bar{w}_i negate each other's changes, and \bar{y}_{i+1} is only scaled by $1/(1 + \epsilon\beta)$, so equality must still hold.

Finally, suppose $w_i \in A_{\ell+1}$. Then $y_i/u_i > w_i/v_i = y_{i+1}/u_{i+1}$, so $y_0 = y_i/u_i$ by tightness of the third constraint and $y_i \notin J$. Both w_i and y_{i+1} decrease in the same direction, so $\bar{w}_i/v_i = \bar{y}_{i+1}/u_{i+1}$ and are smaller than \bar{y}_i/u_i . For the third constraint, \bar{y}_{i+1} and \bar{w}_i negate each other's changes, and \bar{y}_i is only scaled by $1/(1 + \epsilon\beta)$, so equality must still hold.

Case 4 - $\bar{y}_i \in A_\ell$ and $\bar{y}_{i+1} \in A_{\ell+1}$ for even $\ell \neq 0$: A similar analysis as case 3 holds.

Hence $(\bar{\mathbf{y}}, \bar{\mathbf{w}})$ is feasible to Problem (3.7). We can define a symmetric solution $(\mathbf{y}', \mathbf{w}')$ as:

$$y'_i = \begin{cases} \frac{1}{1-\epsilon\beta}y_i & \text{if } y_i \notin J \\ \frac{1}{1-\epsilon\beta}(y_i - \epsilon u_i) & \text{if } y_i \in J \cap A_\ell \text{ for odd } \ell \\ \frac{1}{1-\epsilon\beta}(y_i + \epsilon u_i) & \text{if } y_i \in J \cap A_\ell \text{ for even } \ell \neq 0 \end{cases},$$

$$w'_i = \begin{cases} \frac{1}{1-\epsilon\beta}w_i & \text{if } w_i \notin J \text{ or } w_i \in A_0 \\ \frac{1}{1-\epsilon\beta}(w_i - \epsilon v_i) & \text{if } w_i \in J \cap A_\ell \text{ for odd } \ell \\ \frac{1}{1-\epsilon\beta}(w_i + \epsilon v_i) & \text{if } w_i \in J \cap A_\ell \text{ for even } \ell \neq 0 \end{cases},$$

$$\bar{y}_0 = y_0/(1 - \epsilon\beta).$$

The same case analysis shows that $(\mathbf{y}', \mathbf{w}')$ is feasible to Problem (3.7).

Finally, we have $(\mathbf{y}, \mathbf{w}) = \lambda(\bar{\mathbf{y}}, \bar{\mathbf{w}}) + (1 - \lambda)(\mathbf{y}', \mathbf{w}')$ by taking $\lambda = (1 + \epsilon\beta)/2$. Hence, (\mathbf{y}, \mathbf{w}) is not an extreme point, and any extreme point cannot have y_i such that $0 < y_i/u_i < y_0$. \square

Using Lemma 11, we can now prove Theorem 5.

Proof of Theorem 5. The third part of condition 1 is satisfied by Lemma 11, so we

focus on the first and second parts. If $y_i = 0$ or $y_{i+1} = 0$, then it is clear that $w_i = 0$. Otherwise, $y_i/u_i = y_{i+1}/u_{i+1} = y_0$. Feasibility to the first three constraints of Problem (3.7) ensures that $w_i/v_i = y_i/u_i = y_{i+1}/v_{i+1}$. \square

B.3 Proofs Omitted from Chapter 3

Proof of Claim 2. We only prove the first case because the other two cases are similar. Let \mathbf{x}^* be an optimal assortment with expected revenue strictly greater than δ :

$$\frac{\sum_{i=1}^n \pi_i u_i x_i^* + \sum_{i=1}^{n-1} \sum_{j=i+1}^n \bar{\pi}_{i,j} v_{i,j} x_i^* x_j^*}{1 + \sum_{i=1}^n u_i x_i^* + \sum_{i=1}^{n-1} \sum_{j=i+1}^n v_{i,j} x_i^* x_j^*} > \delta.$$

We can multiply both sides by the denominator because it is strictly positive, and rearrange terms to obtain:

$$\sum_{i=1}^n (\pi_i - \delta) u_i x_i^* + \sum_{i=1}^{n-1} \sum_{j=i+1}^n (\bar{\pi}_{i,j} - \delta) v_{i,j} x_i^* x_j^* > \delta.$$

The left side of the above inequality is upper-bounded by $h(\delta)$. \square

Proof of Theorem 3. Assortment optimization under synergistic MNL is in NP: given an assortment x and some K , it is easy to check whether $\Pi(\mathbf{x}) \geq K$. We prove that the problem is NP-hard via a reduction from the maximum independent set problem.

Given a graph $G = (V, E)$, we need to determine if there exists a subset of vertices $S \subseteq V$ of size K , such that $(i, j) \notin E$ if $i, j \in S$. Index the vertices by $\{1, \dots, n\}$. We can create a synergistic MNL instance where the expected profit is greater or equal to $K/(1 + K)$ if and only if there exists an independent set of size K .

Create a product i for each vertex $i \in V$. We slightly abuse notation and use the same name for a product and its corresponding vertex. Product i has profit $\pi_i = 0$ and base preference weight $u_i = 0$. For the synergy terms, let $v_j^i = v_i^j = n/2$ if $(i, j) \in E$ and $v_j^i = v_i^j = 0$ otherwise. Then $\bar{\pi}_{i,j} = 0$, $v_{i,j} = n$ if $(i, j) \in E$, and $\bar{\pi}_{i,j} = 0$, $v_{i,j} = 0$ if $(i, j) \notin E$.

Introduce an auxiliary product $n + 1$, which has profit $\pi_{n+1} = 1$ and base preference weight $u_{n+1} = 0$. Product $n + 1$ creates synergy with all the other products. Specifically, $v_{n+1}^i = 1$ and $v_i^{n+1} = 0$ for all $i = 1, \dots, n$, so that the preference weight of product $n + 1$ increases by 1 for every additional product offered. Then $\bar{\pi}_{i,n+1} = 1$ and $v_{i,n+1} = 1$. Since this is the only product with non-zero profit, it must be included in an optimal assortment.

We may disregard $S = \emptyset$ as a single vertex is always an independent set. For a set $S \subseteq V$, create an assortment \mathbf{x} where $x_i = \mathbb{1}[i \in S]$ for $i = 1 \leq i \leq n$ and $x_{n+1} = 1$. Then the expected profit is:

$$\begin{aligned} \Pi(\mathbf{x}) &= \frac{\sum_{i=1}^n \bar{\pi}_{i,n+1} x_i}{1 + \sum_{(i,j) \in E} v_{i,j} x_i x_j + \sum_{i=1}^n v_{i,n+1} x_i} \\ &= \frac{\sum_{i=1}^n x_i}{1 + n \sum_{(i,j) \in E} x_i x_j + \sum_{i=1}^n x_i}. \end{aligned}$$

First we show that any optimal assortment must correspond to an independent set in G . If $S \neq \emptyset$ is an independent set, then

$$\Pi(\mathbf{x}) = \frac{|S|}{1 + |S|} \geq \frac{1}{2}.$$

If S is not an independent set, then there is at least one edge (i, j) where $x_i = x_j = 1$

$$\Pi(\mathbf{x}) \leq \frac{|S|}{1 + n + |S|} < \frac{1}{2}.$$

The optimal assortment must be an independent set S along with product $n + 1$, with expected profit $|S|/(|S| + 1)$. As the expected profit is increasing in the size of the independent set, there exists an independent set of size K if and only if the optimal expected profit is at least $K/(K + 1)$. \square

Proof of Lemma 3. Suppose we know the value of δ^* and solved Problem (3.5) with $\delta = \delta^*$. By Claim 2, Problem (3.5) has optimal solution V^* with $V_1^* = h(\delta^*) = \delta^*$. Hence V^* is feasible to Problem (3.6) with objective value δ^* .

Now suppose by contradiction that Problem (3.6) has optimal solution \bar{V} with optimal objective value $\bar{V}_1 = \bar{\delta} < \delta^*$. Then \bar{V} is a feasible solution to Problem (3.5) when $\delta = \bar{\delta}$, with objective value equal to $\bar{\delta}$. But the optimal objective value of Problem (3.5) at $\delta = \bar{\delta}$ is $h(\bar{\delta})$, which implies that $h(\bar{\delta}) \leq \bar{\delta} < \delta^*$. This contradicts part (iii) of Claim 2. \square

Proof of Lemma 4. Construct a solution (\mathbf{y}, \mathbf{w}) from assortment x as follows:

$$\begin{aligned} y_0 &= \frac{1}{1 + \sum_{j=1}^n u_j x_j + \sum_{j=1}^{n-1} v_j x_j x_{j+1}}, \\ y_i &= \frac{u_i x_i}{1 + \sum_{j=1}^n u_j x_j + \sum_{j=1}^{n-1} v_j x_j x_{j+1}}, & i = 1, \dots, n, \\ w_i &= \frac{v_i x_i x_{i+1}}{1 + \sum_{j=1}^n u_j x_j + \sum_{j=1}^{n-1} v_j x_j x_{j+1}}, & i = 1, \dots, n-1. \end{aligned}$$

Then (\mathbf{y}, \mathbf{w}) is feasible to Problem (3.7), satisfies condition 1, and has objective value $\Pi(\mathbf{x})$. \square

Proof of Lemma 5. Since condition 1 is satisfied, we can rewrite the fourth constraint of Problem (3.7) as:

$$1 = y_0 + \sum_{i=1}^n y_0 u_i \cdot \mathbb{1}[y_i > 0] + \sum_{i=1}^{n-1} y_0 v_i \cdot \mathbb{1}[w_i > 0].$$

Condition 1 also tells us that

$$\mathbb{1}[w_i > 0] = \mathbb{1}[y_i > 0] \cdot \mathbb{1}[y_{i+1} > 0].$$

Since $x_i = \mathbb{1}[y_i > 0]$, we can solve for y_0 in terms of \mathbf{x} :

$$y_0 = \frac{1}{1 + \sum_{i=1}^n u_i x_i + \sum_{i=1}^{n-1} v_i x_i x_{i+1}}.$$

Using the same logic and substituting in the value of y_0 , the objective value of (\mathbf{y}, \mathbf{w}) can be rewritten in terms of \mathbf{x} :

$$\begin{aligned} & \sum_{i=1}^n \pi_i y_i + \sum_{i=1}^{n-1} \bar{\pi}_i w_i \\ &= \sum_{i=1}^n \pi_i y_0 u_i \cdot \mathbb{1}[y_i > 0] + \sum_{i=1}^{n-1} \bar{\pi}_i y_0 v_i \cdot \mathbb{1}[w_i > 0] \\ &= \frac{\sum_{i=1}^n \pi_i u_i x_i + \sum_{i=1}^{n-1} \bar{\pi}_i v_i x_i x_{i+1}}{1 + \sum_{i=1}^n u_i x_i + \sum_{i=1}^{n-1} v_i x_i x_{i+1}} = \Pi(\mathbf{x}). \end{aligned}$$

□

APPENDIX C

APPENDIX: ASSORTMENT OPTIMIZATION WITH DYNAMIC SUBSTITUTION AND INVENTORY STOCKING COSTS

C.1 Proofs Omitted from Chapter 4

Proof of Theorem 6. Assortment optimization with dynamic substitution under the deterministic setting is in NP: given a stocking level \mathbf{u} , we can determine whether the retailer's profit is greater or equal to Π for any Π . We show that our problem is NP-hard via a reduction from the maximum independent set problem. In other words, we show that if for any instance of our problem and any Π , we can determine if the optimal profit is at least Π , then we can solve the maximum independent set problem.

In the maximum independent set problem, we are given a graph $G = (V, E)$ with $n = |V|$ and $m = |E|$. We want to determine if there exists an independent set S of size greater or equal to K . An independent set S is defined to be $S \subseteq V$ such that none of the vertices in S are adjacent to each other. That is, for all pairs $i, j \in S$, there are no edges $(i, j) \in E$. Let d_i represent the degree of vertex i , and $D = \max_{i \in V} d_i + 1$.

We construct an instance of our assortment optimization problem with solution \mathbf{u} that achieves a profit of $\Pi = nD + K$ if and only if the maximum independent set of G has size K . The retailer has access to $n + 1$ products. Let $N = \{1, \dots, n, n + 1\}$. Products $i \in N \setminus \{n + 1\}$ have profits $\pi_i = (D + 1)/(d_i + 1)$. Product $n + 1$ has profit $\pi_{n+1} = D$. There are $T = n + m$ customers arriving over the selling horizon. The first n customers are vertex customers. Each vertex cus-

customer $t = i \leq n$ considers only products i and $n + 1$ and ranks them as $i \succ_t n + 1$. Customers $t = n + 1$ through $n + m$ are called edge customers. An edge customer considers products i and j corresponding to the two vertices incident to her edge, with preference list $i \succ_t j$ where $i < j$. The edge customers can arrive in any order. Notice that for each $i \in N \setminus \{n + 1\}$, there are exactly $d_i + 1$ customers who have product i in their consideration set: one vertex customer and d_i edge customers.

First, we show that if we are given an independent set S on G , then we can construct \mathbf{u} which achieves profit of $nD + |S|$. If $i \in S$, then set $u_i = d_i + 1$. If $i \notin S$, then set $u_i = 0$. Finally, set $u_{n+1} = n - |S|$. For product $i \in S$, there is sufficient quantity of product i to satisfy every customer who considers it. Vertex customer $t = i$ will purchase product i over $n + 1$, and any edge customer $e = (i, j)$ must purchase product i since $j \notin S$. Hence all u_i units are sold and the profit from selling product i is:

$$(d_i + 1) \cdot \pi_i = (d_i + 1) \cdot \frac{D + 1}{d_i + 1} = D + 1.$$

For product $i \notin S$, vertex customer $t = i$ will purchase product $n + 1$ because there are sufficient units of product $n + 1$ available. The profit from selling product $n + 1$ is:

$$(n - |S|) \cdot \pi_{n+1} = (n - |S|) \cdot D.$$

Hence the total profit is $nD + |S|$.

Next, we show that we can recover a maximum independent set from an optimal assortment and stocking level. Let \mathbf{u}^* be an optimal assortment's stocking level, and let $S = \{i \in N : u_i^* \neq 0\}$. We claim \mathbf{u}^* satisfies two properties:

1. For products $i \in N \setminus \{n + 1\}$, either $u_i^* = 0$ or $u_i^* = d_i + 1$. For product $n + 1$,

$$u_{n+1}^* = n - |S|.$$

2. S is an independent set.

To prove the first property, we start by considering the stocking level of product $n + 1$. A vertex customer $t = i$ purchases product $n + 1$ if and only if $u_i^* = 0$ (i.e. $i \notin S$). If $u_i^* > 0$, then vertex customer $t = i$ is the first customer to request product i and always obtains her first-choice product. An edge customer never purchases product $n + 1$ because product $n + 1$ is not part of her preference list. Hence, the stocking level of product $n + 1$ is $u_{n+1}^* = n - |S|$.

Now suppose by contradiction that there exists product $j \in N \setminus \{n + 1\}$ where $0 < u_j^* < d_j + 1$, and let i be the largest such index. That is, $i = \arg \max\{j \in N \setminus \{n + 1\} : 0 < u_j^* < d_j + 1\}$. Product i is purchased by the vertex customer $t = i$, and possibly by some edge customers $t = (i, j)$. The profit from selling product i is at most:

$$\begin{aligned} u_i^* \cdot \pi_i &\leq d_i \cdot \frac{D + 1}{d_i + 1} \\ &= \frac{Dd_i + d_i + D - D}{d_i + 1} \\ &= D - \frac{D - d_i}{d_i + 1} \\ &< D. \end{aligned}$$

The last line is true because $d_i < D$ for all $i \in N \setminus \{n + 1\}$. Instead, we can increase u_{n+1}^* by one unit and set $u_i^* = 0$. Then vertex customer $t = i$ purchases product $n + 1$ instead of product i for a profit of D . Any edge customers $t = (i, j)$ who previously purchased product i will now leave without a purchase. To see why this holds, suppose that $j < i$. Then $j \succ_t i$ and customer t purchased product i because product j has stocked out or was never in the assortment. Hence customer t leaves without a purchase under the revised \mathbf{u}^* . Otherwise, if $j >$

i , then customer t has preference list $i \succ_t j$. Since customer t had previously purchased product i , we know that $u_j^* < d_j + 1$. By maximality of index i , we must have $u_j^* = 0$ and customer t cannot purchase product j . Hence, the net change in profit from dropping all units of product i and adding one unit of product $n + 1$ is strictly positive, and contradicts the optimality of \mathbf{u}^* .

To prove the second property, suppose by contradiction that there exists $i, j \in S$ such that $(i, j) \in E$. Since $i, j \in S$, we know $u_i^* = d_i + 1$ and $u_j^* = d_j + 1$ by the first property. The corresponding edge customer purchases one of products i and j , and not all units of the other product are sold. This contradicts the condition that all units are sold in the deterministic setting. Hence S must be an independent set.

Since S is an independent set and the assortment built from an independent set has profit $nD + |S|$, we can find a maximum independent set of size K if and only if we can find an assortment which generates a profit of $\Pi = nD + K$. \square

Proof of Lemma 6. First, we show that every feasible solution of Problem (4.2) can be used to construct a feasible solution to Problem (4.1) with equal objective value. Suppose that (\mathbf{x}, \mathbf{y}) is feasible to Problem (4.2). Create vectors $\mathbf{x}' \in \mathbb{R}^{nT}$ and $\mathbf{u}' \in \mathbb{R}^n$ such that $\mathbf{x}' = \mathbf{x}$ and $u'_i = \sum_{t=1}^T x_{t,i}$. Then the objective values are equal by definition of \mathbf{u}' . As previously argued, we know that $\sum_{i=1}^n x'_{t,i} \leq 1$ for $t = 1, \dots, T$. The only constraint which we need to check is the third one: $u'_j - \sum_{t'=1}^{t-1} x'_{t',j} \leq T \cdot (1 - x'_{t,i})$ for $j \succ_t i$.

Suppose $u'_j - \sum_{t'=1}^{t-1} x'_{t',j} = 0$, then this constraint is feasible for both $x'_{t,i} \in \{0, 1\}$. On the other hand, suppose $u'_j - \sum_{t'=1}^{t-1} x'_{t',j} > 0$. Then there exists $x_{s,j} = 1$ for $s \geq t$. That is, there exists a customer who had purchased product j at a later

period $s \geq t$. By feasibility to Problem (4.2), we know $y_{t,j} \geq y_{s,j} \geq x_{s,j} = 1$. Hence $x_{t,i} + y_{t,j} \leq 1$ and this implies that $x'_{t,i} = x_{t,i} = 0$. Thus feasibility to our constraint holds.

Next, we show that every feasible solution of Problem (4.1) can be used to construct a feasible solution to Problem (4.2). Given a feasible solution (\mathbf{x}, \mathbf{u}) to Problem (4.1), define $\tau_i = \min\{t : \sum_{t'=1}^t x_{t'i} = u_i\}$. We construct a solution $\mathbf{x}' \in \mathbb{R}^{nT}$ and $\mathbf{y}' \in \mathbb{R}^{nT}$ to Problem (4.2) by setting $\mathbf{x}' = \mathbf{x}$ and $y'_{t,i} = \mathbb{1}[t \leq \tau_i]$. The objective values are equal because our solution to Problem (4.1) satisfies $\sum_{t=1}^T x_{t,i} = u_i$. The constraints that we need to check are those that involve \mathbf{y}' .

The monotonicity property is clearly satisfied: $y'_{t,i} = \mathbb{1}[t \leq \tau_i] \geq \mathbb{1}[t+1 \leq \tau_i] = y'_{t+1,i}$. For the second constraint related to \mathbf{y}' , consider the case where $t \leq \tau_i$ and $t > \tau_i$. If $t \leq \tau_i$, then $y'_{t,i} = 1 \geq x'_{t,i}$ regardless of whether $x'_{t,i} \in \{0, 1\}$. If $t > \tau_i$, then $y'_{t,i} = 0$. By feasibility to Problem (4.1) and definition of τ_i , we have that:

$$\sum_{t'=1}^{\tau_i} x_{t'i} = u_i = \sum_{t'=1}^T x_{t'i}.$$

This implies that $x'_{t,i} = 0$ for all $t > \tau_i$ and hence $x'_{t,i} \leq y'_{t,i}$ is always satisfied. For the third constraint related to \mathbf{y}' , we want to show that $y'_{t,j} + x'_{t,i} \leq 1$ whenever $j \succ_t i$. If $y'_{t,j} = 0$, then this is clearly satisfied. On the other hand, suppose $y'_{t,j} = 1$. Then $t \leq \tau_j$, and by definition of τ_j , we have $u_j - \sum_{t'=1}^{t-1} x_{t'j} > 0$. By feasibility to Problem (4.1), we have $x_{t,i} = 0$ and hence $x'_{t,i} = 0$. Thus, $(\mathbf{x}', \mathbf{y}')$ is feasible to Problem (4.2). \square

Proof of Lemma 7. Let (\mathbf{x}, \mathbf{y}) be a basic optimal solution to the linear programming relaxation of Problem (4.2). We first observe that at an optimal solution, we must have $x_{t,i} = \min\{y_{t,i} \cup \{1 - y_{t,j} : j \succ_t i\}\}$ for all $i \in \sigma_t$. Hence, if (\mathbf{x}, \mathbf{y}) is not half-integral, then there must exist some $y_{t,i} \notin \{0, 1/2, 1\}$. Let

$S^+ = \{(t, i) : 1/2 < y_{t,i} < 1\}$, and $S^- = \{(t, i) : 0 < y_{t,i} < 1/2\}$. We define two new feasible solutions, $(\mathbf{x}^1, \mathbf{y}^1)$ and $(\mathbf{x}^2, \mathbf{y}^2)$ such that $(\mathbf{x}, \mathbf{y}) = 1/2 \cdot ((\mathbf{x}^1, \mathbf{y}^1) + (\mathbf{x}^2, \mathbf{y}^2))$. For some small $\epsilon > 0$, define vectors \mathbf{y}^1 and \mathbf{y}^2 as:

$$y_{t,i}^1 = \begin{cases} y_{t,i} + \epsilon & : (t, i) \in S^+ \\ y_{t,i} - \epsilon & : (t, i) \in S^- \\ y_{t,i} & : (t, i) \notin S^+ \cup S^- \end{cases},$$

$$y_{t,i}^2 = \begin{cases} y_{t,i} - \epsilon & : (t, i) \in S^+ \\ y_{t,i} + \epsilon & : (t, i) \in S^- \\ y_{t,i} & : (t, i) \notin S^+ \cup S^- \end{cases}.$$

Finally, let $x_{t,i}^1 = \min\{y_{t,i}^1\} \cup \{1 - y_{t,j}^1 : j \succ_t i\}$ and $x_{t,i}^2 = \min\{y_{t,i}^2\} \cup \{1 - y_{t,j}^2 : j \succ_t i\}$. We can also define $S^{+1} = \{(t, i) : 1/2 < y_{t,i}^1 < 1\}$ and $S^{-1} = \{(t, i) : 0 < y_{t,i}^1 < 1/2\}$. We define S^{+2}, S^{-2} similarly based on \mathbf{y}^2 . For small enough $\epsilon > 0$, $S^+ = S^{+1} = S^{+2}$ and $S^- = S^{-1} = S^{-2}$.

Under this definition, it is clear that $y_{t,i} = 1/2 \cdot (y_{t,i}^1 + y_{t,i}^2)$. We focus on proving $x_{t,i} = 1/2 \cdot (x_{t,i}^1 + x_{t,i}^2)$.

Case 1 ($x_{t,i} = 0$): Then either $y_{t,i} = 0$ or there exists $j \succ_t i$ such that $y_{t,j} = 1$. We have $y_{t,i}^1 = y_{t,i}^2 = 0$ in the former case and $y_{t,j}^1 = y_{t,j}^2 = 1$ in the latter case. Hence $x_{t,i}^1 = x_{t,i}^2 = 0$.

Case 2 ($x_{t,i} = 1$): We have $y_{t,i} = 1$ and $y_{t,j} = 0$ for all $j \succ_t i$. Hence $y_{t,i}^1 = y_{t,i}^2 = 1$ and $y_{t,j}^1 = y_{t,j}^2 = 0$, so $x_{t,i}^1 = x_{t,i}^2 = 1$.

Case 3 ($x_{t,i} = 1/2$): Then $y_{t,i} \geq 1/2$ and $y_{t,j} \leq 1/2$ for all $j \succ_t i$, with at least one of these terms being exactly equal to $1/2$. Since ϵ is small enough, $(t, i) \in S^{+1} \cap S^{+2}$ if $(t, i) \in S^+$, and $(t, j) \in S^{-1} \cap S^{-2}$ if $(t, j) \in S^-$. The term equal to $1/2$ remains

unchanged. Hence, $\min\{\{y_{t,i}^1\} \cup \{1 - y_{t,j}^1 : j \succ_t i\}\} = 1/2$ and $x_{t,i}^1 = 1/2$. The same argument holds for $x_{t,i}^2 = 1/2$.

Case 4 ($0 < x_{t,i} < 1/2$): We can observe:

$$\begin{aligned} x_{t,i} &= \min\{\{y_{t,i}\} \cup \{1 - y_{t,j} : j \succ_t i\}\} \\ &= \min\{\{y_{t,i} : (t,i) \in S^-\} \cup \{1 - y_{t,j} : j \succ_t i, (t,j) \in S^+\}\}. \end{aligned}$$

We focus on the second line. The union of the two sets is non-empty because $0 < x_{t,i} < 1/2$. Furthermore, for small enough ϵ , this equality holds for both $x_{t,i}^1$ and $x_{t,i}^2$ because $S^- = S^{-1} = S^{-2}$ and $S^+ = S^{+1} = S^{+2}$. On the right, all the terms decrease by $-\epsilon$ in \mathbf{y}^1 and increase by $+\epsilon$ in \mathbf{y}^2 . Their ordering is preserved and hence:

$$\begin{aligned} x_{t,i}^1 &= \min\{\{y_{t,i}^1 : (t,i) \in S^{-1}\} \cup \{1 - y_{t,j}^1 : j \succ_t i, (t,j) \in S^{+1}\}\} \\ &= \min\{\{y_{t,i} : (t,i) \in S^-\} \cup \{1 - y_{t,j} : j \succ_t i, (t,j) \in S^+\}\} - \epsilon = x_{t,i} - \epsilon, \\ x_{t,i}^2 &= \min\{\{y_{t,i}^2 : (t,i) \in S^{-2}\} \cup \{1 - y_{t,j}^2 : j \succ_t i, (t,j) \in S^{+2}\}\} \\ &= \min\{\{y_{t,i} : (t,i) \in S^-\} \cup \{1 - y_{t,j} : j \succ_t i, (t,j) \in S^+\}\} + \epsilon = x_{t,i} + \epsilon. \end{aligned}$$

Case 5 ($1/2 < x_{t,i} < 1$): It must be true that $(t,i) \in S^+$ and $(t,j) \in S^-$ for all $j \succ_t i$, because $1/2 < \min\{\{y_{t,i}\} \cup \{1 - y_{t,j} : j \succ_t i\}\} < 1$. All of the terms in $\min\{\{y_{t,i}\} \cup \{1 - y_{t,j} : j \succ_t i\}\}$ increase by $+\epsilon$ in \mathbf{y}^1 and decrease by $-\epsilon$ in \mathbf{y}^2 . Their ordering is preserved and hence:

$$\begin{aligned} x_{t,i}^1 &= \min\{\{y_{t,i}^1\} \cup \{1 - y_{t,j}^1 : j \succ_t i\}\} = \min\{\{y_{t,i}\} \cup \{1 - y_{t,j} : j \succ_t i\}\} + \epsilon = x_{t,i} + \epsilon, \\ x_{t,i}^2 &= \min\{\{y_{t,i}^2\} \cup \{1 - y_{t,j}^2 : j \succ_t i\}\} = \min\{\{y_{t,i}\} \cup \{1 - y_{t,j} : j \succ_t i\}\} - \epsilon = x_{t,i} - \epsilon. \end{aligned}$$

In all five cases above, we have $x_{t,i} = 1/2 \cdot (x_{t,i}^1 + x_{t,i}^2)$. Hence, a basic optimal solution must be half-integral. \square

BIBLIOGRAPHY

- Albadvi, A. and Shahbazi, M. (2009), 'A hybrid recommendation technique based on product category attributes', *Expert Systems with Applications* **36**(9), 11480–11488.
- Anderson, S. P., De Palma, A. and Thisse, J. F. (1992), *Discrete choice theory of product differentiation*, MIT press.
- Aouad, A., Farias, V. F. and Levi, R. (2015), 'Assortment optimization under consider-then-choose choice models'.
- Aouad, A., Farias, V. F., Levi, R. and Segev, D. (2015), 'The approximability of assortment optimization under ranking preferences'.
- Aouad, A., Levi, R. and Segev, D. (2015a), 'Approximation algorithms for dynamic assortment optimization models'.
- Aouad, A., Levi, R. and Segev, D. (2015b), 'Greedy-like algorithms for dynamic assortment planning under multinomial logit preferences'.
- Avery, J., Steenburgh, T. J., Deighton, J. and Caravella, M. (2012), 'Adding bricks to clicks: Predicting the patterns of cross-channel elasticities over time', *Journal of Marketing* **76**(3), 96–111.
- Bachrach, D. G., Ogilvie, J., Rapp, A. and Calamusa IV, J. (2016), *More Than a Showroom: Strategies for Winning Back Online Shoppers*, Springer.
- Balakrishnan, A., Sundaresan, S. and Zhang, B. (2014), 'Browse-and-switch: Retail-online competition under value uncertainty', *Production and Operations Management* **23**(7), 1129–1145.

- Barahona, F. (1986), 'A solvable case of quadratic 0–1 programming', *Discrete Applied Mathematics* **13**(1), 23–26.
- Bell, D., Gallino, S. and Moreno, A. (2014), 'How to win in an omnichannel world', *MIT Sloan Management Review* **56**(1), 45.
- Bell, D., Gallino, S. and Moreno, A. (2015), 'Showrooms and information provision in omni-channel retail', *Production and Operations Management* **24**(3), 360–362.
- Bell, D., Gallino, S. and Moreno, A. (2017), 'Offline showrooms in omnichannel retail: Demand and operational benefits', *Management Science* .
- Ben-Akiva, M. E. (1973), 'The structure of travel demand models', *PhD Thesis* .
- Ben-Akiva, M. E. and Lerman, S. R. (1985), *Discrete choice analysis: theory and application to travel demand*, Vol. 9, MIT press.
- Blanchet, J., Gallego, G. and Goyal, V. (2016), 'A markov chain approximation to choice modeling', *Operations Research* **64**(4), 886–905.
- Blue Nile (2018), 'FAQs: About Blue Nile', <https://www.bluenile.com/contact-us/faq>. Accessed: 2018-05-29.
- Bodlaender, H. L. (1996), 'A linear-time algorithm for finding tree-decompositions of small treewidth', *SIAM Journal on computing* **25**(6), 1305–1317.
- Bodlaender, H. L., Gilbert, J. R., Hafsteinsson, H. and Kloks, T. (1995), 'Approximating treewidth, pathwidth, frontsize, and shortest elimination tree', *J. Algorithms* **18**(2), 238–255.
- Brandstadt, A., Spinrad, J. P. et al. (1999), *Graph classes: a survey*, Vol. 3, Siam.

- Bront, J. J. M., Méndez-Díaz, I. and Vulcano, G. (2009), 'A column generation algorithm for choice-based network revenue management', *Operations Research* **57**(3), 769–784.
- Cho, Y. H., Kim, J. K. and Kim, S. H. (2002), 'A personalized recommender system based on web usage mining and decision tree induction', *Expert systems with Applications* **23**(3), 329–342.
- Davis, J. M., Gallego, G. and Topaloglu, H. (2013), 'Assortment planning under the multinomial logit model with totally unimodular constraint structures', *Department of IEOR, Columbia University. Available at http://www.columbia.edu/gmg2/logit_const.pdf*.
- Davis, J. M., Gallego, G. and Topaloglu, H. (2014), 'Assortment optimization under variants of the nested logit model', *Operations Research* **62**(2), 250–273.
- Désir, A., Goyal, V. and Zhang, J. (2014), 'Near-optimal algorithms for capacity constrained assortment optimization'.
- Dzyabura, D. and Jagabathula, S. (2017), 'Offline assortment optimization in the presence of an online channel', *Management Science*.
- Feldman, J. B., Paul, A. and Topaloglu, H. (2017), 'Assortment optimization with small consideration sets', *Available at SSRN 3013357*.
- Feldman, J. B. and Topaloglu, H. (2014), 'Revenue management under the markov chain choice model', *Submitted for publication*.
- Feldman, J. B. and Topaloglu, H. (2015), 'Capacity constraints across nests in assortment optimization under the nested logit model', *Operations Research* **63**(4), 812–822.

- Feldman, J. B. and Topaloglu, H. (2017), 'Revenue management under the markov chain choice model', *Operations Research* **65**(5), 1322–1342.
- Feldman, J. B., Zhang, D., Liu, X. and Zhang, N. (2018), 'Taking assortment optimization from theory to practice: Evidence from large field experiments on alibaba'.
- Feng, G., Li, X. and Wang, Z. (2015), 'Analysis of discrete choice models: A welfare-based framework', *arXiv preprint arXiv:1503.01854*.
- Fornari, E., Fornari, D., Grandi, S., Menegatti, M. and Hofacker, C. F. (2016), 'Adding store to web: migration and synergy effects in multi-channel retailing', *International Journal of Retail & Distribution Management* **44**(6), 658–674.
- Gallego, G., Iyengar, G., Phillips, R. and Dubey, A. (2004), 'Managing flexible products on a network'.
- Gallego, G., Ratliff, R. and Shebalov, S. (2014), 'A general attraction model and sales-based linear program for network revenue management under customer choice', *Operations Research* **63**(1), 212–232.
- Gangurde, S. R. and Akarte, M. M. (2015), 'Segmentation based product design using preferred features', *Benchmarking* **22**(6), 1096–1114.
- Gao, F. and Su, X. (2016a), 'Omnichannel retail operations with buy-online-and-pick-up-in-store', *Management Science* **63**(8), 2478–2492.
- Gao, F. and Su, X. (2016b), 'Online and offline information for omnichannel retailing', *Manufacturing & Service Operations Management* **19**(1), 84–98.
- Goyal, V., Levi, R. and Segev, D. (2016), 'Near-optimal algorithms for the assort-

- ment planning problem under dynamic substitution and stochastic demand', *Operations Research* **64**(1), 219–235.
- Hanks, A. S., Just, D. R. and Wansink, B. (2012), 'Trigger foods: the influence of "irrelevant" alternatives in school lunchrooms'.
- Honhon, D., Gaur, V. and Seshadri, S. (2010), 'Assortment planning and inventory decisions under stockout-based substitution', *Operations research* **58**(5), 1364–1379.
- Honhon, D., Jonnalagedda, S. and Pan, X. A. (2012), 'Optimal algorithms for assortment selection under ranking-based consumer choice models', *Manufacturing & Service Operations Management* **14**(2), 279–289.
- Huber, J., Payne, J. W. and Puto, C. (1982), 'Adding asymmetrically dominated alternatives: Violations of regularity and the similarity hypothesis', *Journal of consumer research* **9**(1), 90–98.
- Kochenberger, G., Hao, J.-K., Glover, F., Lewis, M., Lü, Z., Wang, H. and Wang, Y. (2014), 'The unconstrained binary quadratic programming problem: a survey', *Journal of Combinatorial Optimization* **28**(1), 58–81.
- Lammers, H. B. (1991), 'The effect of free samples on immediate consumer purchase', *Journal of Consumer Marketing* **8**(2), 31–37.
- Li, G. and Rusmevichientong, P. (2014), 'A greedy algorithm for the two-level nested logit model', *Operations Research Letters* **42**(5), 319–324.
- Li, G., Rusmevichientong, P. and Topaloglu, H. (2015), 'The d-level nested logit model: Assortment and price optimization problems', *Operations Research* **63**(2), 325–342.

- Li, H. and Huh, W. T. (2011), 'Pricing multiple products with the multinomial logit and nested logit models: Concavity and implications', *Manufacturing & Service Operations Management* **13**(4), 549–563.
- Liu, Q. and Van Ryzin, G. (2008), 'On the choice-based linear programming model for network revenue management', *Manufacturing & Service Operations Management* **10**(2), 288–310.
- Luce, R. D. (1959), *Individual Choice Behavior a Theoretical Analysis*, John Wiley and sons.
- Mahajan, S. and Van Ryzin, G. (2001a), 'Inventory competition under dynamic consumer choice', *Operations Research* **49**(5), 646–657.
- Mahajan, S. and Van Ryzin, G. (2001b), 'Stocking retail assortments under dynamic consumer substitution', *Operations Research* **49**(3), 334–351.
- McFadden, D. (1973), 'Conditional logit analysis of qualitative choice behavior'.
- McFadden, D. (1980), 'Econometric models for probabilistic choice among products', *Journal of Business* pp. S13–S29.
- McFadden, D. and Train, K. (2000), 'Mixed MNL models for discrete response', *Journal of applied Econometrics* **15**(5), 447–470.
- Mishra, S., Umesh, U. and Stem Jr, D. E. (1993), 'Antecedents of the attraction effect: An information-processing approach', *Journal of Marketing Research* pp. 331–349.
- Nagarajan, M. and Rajagopalan, S. (2008), 'Inventory models for substitutable products: optimal policies and heuristics', *Management Science* **54**(8), 1453–1466.

- Padberg, M. (1989), 'The boolean quadric polytope: some characteristics, facets and relatives', *Mathematical programming* **45**(1), 139–172.
- Paul, A., Topaloglu, H. and Feldman, J. B. (2016), 'Assortment optimization for choosy customers'.
- Pfaff (1998), 'A bigger, better ikea experience', *HFN the Weekly Newspaper for the Home Furnishing Network* p. 30.
- Radzik, T. (1998), Fractional combinatorial optimization, in 'Handbook of combinatorial optimization', Springer, pp. 429–478.
- Rusmevichientong, P., Shen, Z.-J. M. and Shmoys, D. B. (2009), 'A PTAS for capacitated sum-of-ratios optimization', *Operations Research Letters* **37**(4), 230–238.
- Rusmevichientong, P., Shen, Z.-J. M. and Shmoys, D. B. (2010), 'Dynamic assortment optimization with a multinomial logit choice model and capacity constraint', *Operations research* **58**(6), 1666–1680.
- Rusmevichientong, P., Shmoys, D. B., Tong, C. and Topaloglu, H. (2014), 'Assortment optimization under the multinomial logit model with random choice parameters', *Production and Operations Management* **23**(11), 2023–2039.
- Simonson, I. (1999), 'The effect of product assortment on buyer preferences', *Journal of Retailing* **75**(3), 347–370.
- Smith, S. A. and Agrawal, N. (2000), 'Management of multi-item retail inventory systems with demand substitution', *Operations Research* **48**(1), 50–64.
- Spencer, J. H. (1987), *Ten lectures on the probabilistic method*, Vol. 52, Society for Industrial and Applied Mathematics Philadelphia.

- Talluri, K. and Van Ryzin, G. (2004), 'Revenue management under a general discrete choice model of consumer behavior', *Management Science* **50**(1), 15–33.
- Thurstone, L. L. (1927), 'A law of comparative judgment.', *Psychological review* **34**(4), 273.
- Topaloglu, H. (2013), 'Joint stocking and product offer decisions under the multinomial logit model', *Production and Operations Management* **22**(5), 1182–1199.
- Train, K. (2003), *Discrete choice methods with simulation*, Cambridge university press.
- Van Ryzin, G. and Mahajan, S. (1999), 'On the relationship between inventory costs and variety benefits in retail assortments', *Management Science* **45**(11), 1496–1509.
- Wächter, A. and Biegler, L. T. (2006), 'On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming', *Mathematical programming* **106**(1), 25–57.
- Wald, J. A. and Colbourn, C. J. (1983), 'Steiner trees, partial 2-trees, and minimum IFI networks', *Networks* **13**(2), 159–167.
- Wang, X., Truong, V. and Bank, D. (2015), Online advance admission scheduling for services, with customer preferences, in 'Working paper'.
- Williamson, D. P. and Shmoys, D. B. (2011), *The design of approximation algorithms*, Cambridge university press.

Ziegler, C.-N., Lausen, G. and Schmidt-Thieme, L. (2004), Taxonomy-driven computation of product recommendations, *in* 'Proceedings of the thirteenth ACM international conference on Information and knowledge management', ACM, pp. 406–415.